VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

INFORMAČNÍ SYSTÉM ORGANIZACE POŘÁDAJÍCÍ SKUPINOVÉ POBYTY

INFORMATION SYSTEM FOR AN ORGANIZATION OF GROUP STAYS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **STANISLAV MECHL**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. VLADIMÍR BARTÍK, Ph.D.**
SUPERVISOR

BRNO 2019

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav informačních systémů (UIFS) Akademický rok 2018/2019

Zadání bakalářské práce

Student: **Mechl Stanislav**
Program: Informační technologie
Název: **Informační systém organizace pořádající skupinové pobyty**
Information System For an Organization of Group Stays
Kategorie: Informační systémy

Zadání:

1. Seznamte se s principy tvorby webových aplikací, dostupnými prostředími a frameworky.
2. Analyzujte požadavky na informační systém pro Dům Ignáce Stuchlého, který pořádá adaptační kurzy a jiné skupinové pobyty. Informační systém bude umožňovat přihlašování na různé akce, plánovat personální zajištění těchto akcí, komunikaci s přihlášenými, tvorbou závěrečných zpráv a vyúčtování pobytů.
3. Navrhněte informační systém dle požadavků, použijte k tomu vhodné modelovací techniky.
4. Po konzultaci s vedoucím navržený informační systém implementujte a otestujte funkčnost na vhodném vzorku dat.
5. Zhodnoťte dosažené výsledky a diskutujte další možné pokračování tohoto projektu.

Literatura:

- Naramore, E., Gerner, J. et al: PHP 6, MySQL, Apache: Vytváříme webové aplikace. Computer Press, 2009. ISBN: 978-8-0251-2767-4.
- Žára, O.: JavaScript - Programátorské techniky a webové technologie, Computer Press, 2015. ISBN: 978-80-251-4573-9

Pro udělení zápočtu za první semestr je požadováno:

- Body 1-3.

Podrobné závazné pokyny pro vypracování práce viz http://www.fit.vutbr.cz/info/szz/

Vedoucí práce: **Bartík Vladimír, Ing., Ph.D.**
Vedoucí ústavu: Kolář Dušan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2018
Datum odevzdání: 15. května 2019
Datum schválení: 26. října 2018

Abstrakt

Cílem této bakalářské práce je navrhnout a implementovat informační systém pro Salesiánský klub mládeže, z. s. Dům Ignáce Stuchlého ve Fryštáku (dále jen Dům Ignáce Stuchlého). Jedná se o výchovně vzdělávací zařízení, které pořádá volnočasové aktivity pro mládež. Mezi tyto aktivity patří například tábory, víkendové pobyty a kroužky. Úkolem je navrhnout databázi, která bude evidovat jednotlivé akce a účastníky, a také uživatelské rozhraní umožňující uživatelům se na akce přihlašovat. Na základě těchto dat bude poté možné zjednodušit práci zaměstnanců. Jedná se především o evidenci seznamů přihlášených účastníků, komunikaci s účastníky akcí a přehled o počtu akcí, kterých se konkrétní uživatel zůčastnil. Navíc má systém umožnit evidenci dárců a jejich finančních darů organizaci.

Abstract

The main aim of this bachelor thesis is to design and develop information system for Salesiánský klub mládeže, z. s. Dům Ignáce Stuchlého ve Fryštáku, which is formative educational facility that organizes leisure activities for youth. Some of theese activities are for example summer camps, weekend events and courses. My task was designing database to record events and participators and also developing user interface that enables users to apply for an event. With utilisation of stored data will be possible to make work of employees more efficient. Especially in procedures related to evidence of participators, communication with them and monitoring count of events that one person attended. In addition system should store data about donations from persons and companies.

Klíčová slova

nette, php, ajax, informační systém, databáze, html, css, webová aplikace

Keywords

nette, php, ajax, information system, database, html, css, web application

Citace

MECHL, Stanislav. *Informační systém organizace pořádající skupinové pobyty*. Brno, 2019. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Vladimír Bartík, Ph.D.

Informační systém organizace pořádající skupinové pobyty

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Vladimíra Bartíka Ph.D. Informace týkající se fungování organizace mi poskytli zaměstnanci Salesiánského klubu mládeže, z. s. Dům Ignáce Stuchlého ve Fryštáku. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Stanislav Mechl
16. května 2019

Poděkování

Tímto chci poděkovat panu Ing. Vladimíru Bartíkovi Ph.D. za odborné vedení mé práce. Dále děkuji zaměstnancům Salesiánského klubu mládeže, z. s. Dům Ignáce Stuchlého ve Fryštáku, zvláště Bc. Dagmar Zlámalíkové, za informace týkající se požadavků na informační systém. Také děkuji Bc. Martě Jonáškové a všem, kteří se zapojili do testování výsledné aplikace.

Obsah

Kapitola 1

Úvod

Tématem této bakalářské práce je tvorba informačního systému pro Dům Ignáce Stuchlého ve Fryštáku. Tento dům patří salesiánům, což je řeholní kongregace zaměřující se na práci s mládeží. Jsou zde pořádány například zážitkové víkendy pro mládež, stmelovací kurzy pro školní třídy, letní tábory, kroužky a jiné volnočasové aktivity. Každý typ aktivit má odlišnou cílovou skupinu a tak se na tyto akce přihlašuje široké spektrum účastníků. s celou organizací se poté pojí velká administrativní zátěž na zaměstnance. Ta spočívá mimo jiné v komunikaci s účastníky, kteří mají zájem se na akci přihlásit, a vytváření seznamů potřebných například pro prezenci po příjezdu či kontrolu provedení platby za akci.

Cílem této bakalářské práce bylo vytvořit webovou aplikaci, která umožní účastníkům prohlížet plánované akce a přihlašovat se na ně prostřednictvím jednotného systému. Tyto přihlášky je poté potřeba evidovat, což v informačním systému znamená databázi ukládající data o akcích a uživatelích. Na této databázi je poté možné postavit interní část systému určenou pro zaměstnance. Ti s evidencí účastníků pravidelně pracují. v současné době využívají několik vzájemně nekomunikujících nástrojů. Mezi ty patří například elektronické dotazníky umístěné na externích webových stránkách, žádosti o přihlášení poslané pomocí emailu a evidence v tabulkovém procesoru. Mnoho časově náročných činností týkajících se evidence účastníků a komunikace s nimi lze s využitím databáze a webové aplikace zefektivnit. Zahrnutí potřebných úkonů do jednoho přihlašovacího systému umožní přístup k datům všem zaměstnancům a zjednoduší sdílení dat. Výhodou je také větší kontrola nad osobními údaji přihlašovaných účastníků.

Text této práce se v kapitole 2 věnuje teoretickému základu. v rámci toho jsou popsané základní pojmy a principy týkající se webových aplikací a databází. Kapitola 3 se věnuje zadání této konkrétní aplikace. Je v ní popsáno fungování Domu Ignáce Stuchlého, specifika jednotlivých typů akcí, personální struktura organizace a definice požadavků na informační systém. Na základě těchto požadavků je poté vytvořen návrh specifikace cílového informačního systému a model databáze (3.4). Následující kapitola 4 popisuje obecně programovací jazyky a technologie použité při tvorbě aplikace. v kapitole 5 je poté popsána konkrétní struktura zdrojových kódů této aplikace a detaily řešení jednotlivých částí implementace. Na tuto část navazuje kapitola 6 o testování výsledného informačního systému a kapitola 7 obsahující zhodnocení výsledku a nastiňující další možnosti vývoje systému a jeho budoucích rozšíření.

Kapitola 2

Teoretická část

Na úvod bych se rád věnoval teoretickému základu své bakalářské práce. Zadáním je naprogramovat informační systém. Krátce v této kapitole popíši principy tvorby webových aplikací a pojmy, které s ní souvisí.

2.1 Informační systém

Důležitým pojmem je sousloví *informační systém*, které je použito hned v úvodu názvu této práce. Pro vymezení tohoto pojmu jej nejprve rozdělím na dvě samostatná slova

Informace

Pojem *informace* se v některých publikacích zaměňuje jako synonymum se slovem *data*. Rozdíl v jejich definici je v tom, že *data* jsou uložená v databázi, ale bez znalosti širšího kontextu není zřejmé, co vlastně znamenají. Až na základě znalosti struktury databáze je možné říct, co která položka znamená a získat tak informaci cennou pro uživatele. *Informace* je již pro uživatele smysluplná a tvoří se na základě uložených *dat*.[5]

Podle [3] jsou informace „*data, která prošla zpracováním nebo dostala strukturu dávající jim pro jednotlivce nebo organizaci význam*“. Samotný informační systém tedy pracuje s *daty*, které si ukládá jako jednotlivé samostatné hodnoty. Tyto uložené hodnoty mají v databázi přesně určenou strukturu, díky které je aplikace schopná uložená data opět získat a předat je uživateli ve formě informace k interpretaci.

Systém

Slovník spisovné češtiny definuje *systém* jako „*způsob uspořádání nějakého celku*“ nebo „*soubor jednotlivin navzájem spojených určitou strukturou, sítí vztahů v (uspořádaný) celek*“[22]. Jedná se tedy o komplexní entitu zahrnující do sebe více součástí. v souvislosti s databázemi se podle [3] zahrnuje do databázového systému kromě samotné databáze také systém řízení databáze a kolekce databázových aplikací.

Informační systém do sebe tedy zahrnuje několik součástí od databáze, která ukládá data, až po uživatelské rozhraní, které je zobrazuje uživateli nebo přes které naopak uživatel do systému data vkládá. Jeho definice může být například následující.

„*Podnikový informační systém vytvářejí lidé, kteří prostřednictvím dostupných technologických prostředků a stanovené metodologie zpracovávají podniková data a vytvářejí z nich*

informační a znalostní bázi organizace, sloužící k řízení podnikových procesů, manažerského rozhodování a správě podnikové agendy.“[18]

Informační systém podle této definice netvoří počítač nebo program, ale lidé kteří s ním pracují. Samotná databáze či webová aplikace je prostředkem, který má uživatelům tuto práci zjednodušit a umožňuje ukládat informace do podoby dat. s těmito daty je poté možné v rámci aplikace pracovat získat z nich cenné informace potřebné pro danou organizaci.

V konkrétním případě mého zadání jsou daty především údaje o akcích i jejich účastnících a vedoucích. z těchto dat lze poté získat informace potřebné pro pořádání akce či evidenci účastníků. Informační systém je tvořený databází, se kterou pracují zaměstnanci i účastníci akcí prostřednictvím webové aplikace.

2.2 Databáze

Slovo *databáze* se na prvních stránkách textu vyskytlo již několikrát. Jedná se o velmi důležitou součást této aplikace a také informačních systémů obecně. Nyní tedy popíši, co si pod tímto pojmem lze představit.

Jedná se o úložiště dat, které je společné pro celý systém a může být využíváno současně mnoha uživateli pro odlišné úkoly. Důležitým prvkem je právě uložení dat na jednotném místě. Díky tomu má každý uživatel pracující se systémem k dispozici aktuální informace a po vložení nových dat se změna okamžitě projeví ve všech částech systému. Kromě samotných dat obsahuje databáze tzv. *metadata* obsahující jejich popis a také vztahy mezi jednotlivými daty. Na základě metadat je možné, za předpokladu že požadovaná data nebyla odstraněna, získat správná data i pokud se struktura tabulky změní. Vztahy mezi daty dokáží popsat, zda spolu jednotlivá data souvisí a umožňují získat z jednotlivých dat celky s větší informační hodnotou.[3]

Uložená data jsou takzvaně *perzistentní*. To znamená že jsou stálá a nezávislá na běhu aplikace. která je vytvořila. Ve stejné podobě také musí zůstat i při restartu počítače nebo přerušení dodávky elektrické energie.[21]

2.2.1 Relační model databáze

Relační model je v současnosti nejrozšířenější model používaný pro uložení dat v databází. Na základě matematické teorie množin jej navrhl Edgar Frank Codd. Relační databáze data ukládá ve formě relací, které se uživateli jeví jako tabulky.[5]

Podle [3] jsou prvky relačního modelu následující:

- *Relace - tabulka se sloupci a řádky*
- *Atribut - pojmenovaný sloupec relace*
- *Datová n-tice - řádek relace*
- *Doména - množina přípustných hodnot pro jeden nebo více atributů*
- *Relační databáze - kolekce normalizovaných tabulek*

Každá tabulka má v databázi svůj jednoznačný název stejně jako její jednotlivé sloupce. Data jsou vyhledatelná na základě těchto názvů a nezáleží tedy na uložení tabulek v paměti ani na pořadí sloupců. Stejně tak nezáleží na pořadí záznamů v tabulce.

2.2.2 Integritní omezení

Pro integritní omezení jsou důležité tzv. *klíče*. Jedná se o sloupce tabulky, které splňují zadaná kritéria a pomocí kterých je definován vztah mezi jednotlivými záznamy. Důležité jsou především dva klíče - *primární* a *cizí*.

Pro vytvoření vazby mezi záznamy je nutné mít možnost řádky tabulky jednoznačně identifikovat. Využívá se k tomu hodnota, která je pro každý záznam v dané tabulce unikátní. Pokud je jisté, že se v tabulce stejná hodnota nikdy nebude vyskytovat u více záznamů, můžeme sloupec, případně více sloupců, použít jako primární klíč. Jiný záznam poté může obsahovat odkaz na primární klíč této tabulky, který se nazývá cizí klíč. Nabývat může hodnotu některého z primárních klíčů odkazované tabulky.[21]

Podle počtu záznamů, které jsou do vztahu zahrnuty, je lze rozdělit na vztahy *1:1*, *1:N* a *M:N* [5]. Vztah 1:1 naznačuje, že z obou tabulek je do něj zahrnut právě jeden záznam. u vztahu 1:N je možné jednomu záznamu z tabulky přiřadit libovolné množství záznamů z druhé tabulky. Vztah typu M:N umožňuje přiřazení více hodnot v obou směrech. Například v konkrétním případu vztahu mezi akcemi a účastníky je možné, aby účastník jel na více akcí a naopak na jednu akci aby bylo přihlášených více účastníků.

Integritní omezení lze je udržovat na několika úrovních. [5] jmenuje tyto:

- *Integrita na úrovni tabulky*
- *Integrita na úrovni pole*
- *Integrita na úrovni vztahu*
- *Business pravidla*

Na úrovni tabulky se jedná o omezení týkající se primárního klíče. Tato integrita zajišťuje, že primární klíč bude vždy vyplněný a v rámci dané tabulky jedinečný. Na úrovni pole se jedná o omezení typu případně množiny hodnot, které se mohou v daném sloupci vyskytovat. Integrita na úrovni vztahu zajišťuje správné vyplnění cizích klíčů. Ty musí být buď nevyplněné, nebo obsahovat hodnotu vyskytující se v odkazované tabulce jako primární klíč. Business pravidla přidávají do omezení další vlastnosti vyžadované organizací. Jejich pomocí lze například specifikovat maximální přípustný počet záznamů přiřazených v rámci 1:N relace.[3]

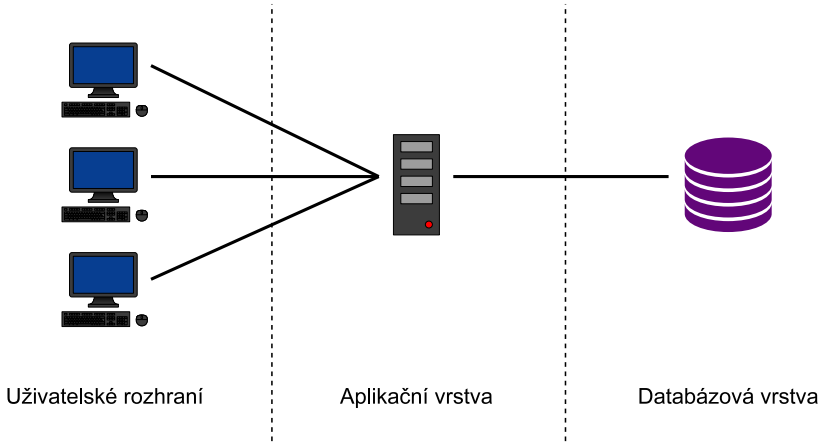
2.3 Webová aplikace

Jazyk SQL usnadňuje práci s databází programátorům, ale pro běžné uživatele je stále nepoužitelný. Je proto nutné vytvořit aplikaci s uživatelským rozhraním, do které bude možné data zadávat a která převede data uložené v databázi na informaci jednoduše čitelnou pro člověka. Tato aplikace může být nainstalovaná lokálně na počítači uživatele nebo běžet jako webová aplikace na serveru. První zmiňovaný přístup převládal dříve, dnes je typičtější využití právě webových aplikací [3]. Jejich výhodou je například možnost přístupu k databázi z jakéhokoli zařízení. Lze tak data upravovat nebo číst také prostřednictvím mobilního telefonu či tabletu.

2.3.1 Architektura klient-server

Celou databázovou aplikaci lze rozdělit na dvě hlavní části. Část aplikace běžící na serveru a část aplikace běžící na počítači uživatele. Před nástupem internetu se většinou na počí-

tači klienta vyskytoval program vykonávající veškerou funkci aplikace a na server posílal databázové dotazy. Tento přístup je nazývaný *tlustý klient*. Komplikací byla složitost instalace na počítače všech uživatelů a nároky na jejich výkon. Později byla tato architektura rozšířena na třívrstvou obsahující vrstvu uživatelského rozhraní, vrstvu zpracovávající data a databázovou vrstvu. Na straně klienta se tak obvykle nachází pouze *tenký klient* v podobě internetového prohlížeče a nevyžaduje tak žádné speciální nainstalované programy.[3]



Obrázek 2.1: Architektura klient-server, převzato z [3]

Databázová vrstva

Pro přístup k samotným datům v databázi se využívá tzv. *systém řízení databáze*. Jedná se o „*softwarový systém, který uživateli umožňuje definovat, vytvářet a udržovat databázi a poskytuje řízený přístup k této databázi*“[3]. Tyto akce s databází jsou prováděné obvykle pomocí dotazovacího jazyka SQL. Mezi systémy řízení databáze se řadí například MySQL[1], z něj odvozený systém MariaDB[2] nebo Microsoft SQL Server[3].

Aplikační vrstva

Zpracování dat má na starosti aplikační vrstva. Ta využívá nejčastěji jazyk PHP. Jedná se o skriptovací jazyk, který byl od základů vytvořen pro potřeby generování webových stránek. Jeho prostřednictví tak lze jednoduše provádět úkony jako formátování HTML kódu, práci s webovými formuláři a komunikaci s databází. Jedná se o multiplatformní open source projekt, což znamená, že funguje na libovolném operačním systému a je zdarma s veřejným zdrojovým kódem.[17]

PHP kód je uložený na straně serveru a ten se také stará o jeho vyhodnocení[19]. v praxi to tedy funguje tak, že uživatel prostřednictvím internetového prohlížeče požádá server o určitou stránku. Server spustí vyhodnocení PHP skriptu. Ten může například zohlednit aktuálního přihlášeného uživatele nebo požádat o data z databáze a jako výsledek vytvoří dokument v HTML. Uživateli je poté serverem odeslána statická internetová stránka.

[1] https://www.mysql.com/
[2] https://mariadb.org/
[3] https://www.microsoft.com/sql-server/

Pro tvorbu aplikací se často využívají *frameworky*. Jedná se o sadu funkcí a tříd, která doplňuje a rozšiřuje funkcionalitu samotného jazyka. Mezi PHP frameworky patří například Laravel[4], Nette[5] a Symfony[6].

Uživatelské rozhraní

Uživatelské rozhraní webových aplikací staví na značkovacím jazyce HTML, který popisuje základní strukturu stránky a obsahuje text zobrazený na stránce. Ten je obvykle doplněn kaskádovými styly, které mají na starosti popis vzhledu dokumentu a formátování.

Dalším prvkem, který se v dnešní době těší velké oblibě, je programovací jazyk Javascript. Základní rozdíl oproti jazyku PHP je ve způsobu vyhodnocení kódu. Zatímco s jazykem PHP pracuje server a ke klientovi se vůbec nedostane, Javascript je naopak odeslán společně s HTML dokumentem a je spuštěn v prohlížeči na klientském počítači, kde umožňuje stránku dynamicky měnit a doplnit ji o interaktivní prvky.[17]

Javascript také dokáže požádat server o konkrétní data a překreslit jen tu část stránky, které se změna týká. Server tak nemusí generovat znovu celou stránku, ale pouze textová data. Tato technologie se nazývá AJAX. Výhodou může být snížení zátěže na server a vyhnutí se zbytečnému načítání částí stránek, které se nemění.

Také pro tuto část aplikace se používají doplňky zjednodušující návrh stránek a poskytující opakovaně využívané prvky. Jedná se například o knihovnu Bootstrap[7] nebo Javascriptové frameworky Angular[8], Vue.js[9] a React[10].

[4] https://laravel.com/
[5] https://nette.org/
[6] https://symfony.com/
[7] https://getbootstrap.com/
[8] https://angular.io/
[9] https://vuejs.org/
[10] https://reactjs.org/

Kapitola 3

Specifikace požadavků

V první části této kapitoly je vysvětleno jak funguje Dům Ignáce Stuchlého a popsány současné postupy a činnosti vykonávané při organizování akcí. Následuje popis jednotlivých typů zde konaných akcí a jejich specifik. v závěru kapitoly je poté vytvořen návrh základních prvků aplikace.

3.1 Dům Ignáce Stuchlého

Dům Ignáce Stuchlého[1] se nachází ve Fryštáku, malém městě poblíž Zlína. Patří řeholní kongregaci salesiánů Dona Boska[2]. Tato kongregace byla založena v 19. století v Itálii. Zakladatelem byl sv. Jan Bosko, kněz věnující se chudým klukům v Turíně. Zaměření na mládež zůstalo salesiánům i do dnešní doby a právě salesiánský dům ve Fryštáku je jedním z míst určených mladým lidem. První salesiáni do Fryštáku dorazili v roce 1927 pod vedením Ignáce Stuchlého, který se následně podílel i na budování dalších salesiánských středisek v České republice [20]. Právě jeho jméno fryštácký dům nese.

Cílová skupina, na kterou se Dům Ignáce Stuchlého zaměřuje, je poměrně široká. Aktivity, které mládeži nabízí, lze rozdělit na pět typů:

- Orientační dny®
- Víkendové akce
- Tábory
- Kroužky
- DISklub

Různé typy akcí se zaměřují na odlišnou skupinu účastníků a spadají do kompetencí jiných zaměstnanců. Počtem účastníků jsou velkou skupinou Orientační dny®[3]. Jedná se o kurzy pro třídní kolektivy, z čehož vyplývá několik specifik. Tím nejdůležitějším je způsob přihlašování, které není otevřeno široké veřejnosti. Je prováděno zástupcem spolupracující školy, který přihlašuje třídu a ne jednotlivé žáky. z tohoto důvodu přihlašování na Orientační dny® není součásti požadavků na tento informační systém.

[1] https://www.disfrystak.cz/
[2] https://www.sdb.cz/
[3] https://www.disfrystak.cz/nase-kurzy/

Na zbývající typy akcí se přihlašuje každý účastník sám za sebe. Základní údaje požadované v přihláškách jsou u všech akcí podobné a jeden účastník se může přihlásit na více různých akcí zároveň. Proto je vhodné, aby v systému byla pro účastníky jen jedna uživatelská role a každý účastník měl stejné možnosti. v následující části vysvětlím specifika jednotlivých typů akcí zahrnutých do požadované aplikace.

Víkendové akce

Prvním typem jsou víkendové akce, pro které se vžil zkrácený název *víkendovky*. Tyto akce jsou jednorázové, obvykle trvají zhruba tři dny a cílovou skupinou je mládež ve věku od 14 let. Účastníci jsou ze širokého okolí a přijíždějí z různých částí České republiky.

Tábory

Letní tábory mají pro reprezentaci v informačním systému podobná specifika jako víkendovky. Jedná se opět o jednorázové akce. Dělí se na dvě skupiny - příměstské a pobytové. Věková kategorie se u jednotlivých táborů může lišit a pohybuje se přibližně v rozmezí 6 až 20 let.

DISklub

Jedná se o volnočasový klub pro mládež z Fryštáku a okolí. Přihláška do klubu platí na celý školní rok a účastník poté může klub navštěvovat kdykoli v otevírací době. Ta je stanovena ve všední dny odpoledne. v DISklubu neprobíhají žádné pravidelné akce.

Kroužky

Zájmové kroužky probíhají pravidelně jednou týdně a jsou přihlašovány na celý školní rok. Je ale potřeba mít možnost v systému zaznamenat ukončení docházky v průběhu. Dalším specifickým údajem je například, zda může účastník odcházet po skončení kroužku samostatně domů.

Vzdělávací akce

Vzdělávací akce jsou specifickou kategorií, která na rozdíl od ostatních není dostupná pro širokou veřejnost. Jsou nabídkou pro lidi, kteří se účastní jiných akcí jako vedoucí či instruktoři. Jedná se opět o jednorázové akce a přihlašování probíhá na každou zvlášť. Zobrazení akcí z této kategorie bude možné pouze pro vybrané uživatele.

Jiné akce

Tato kategorie je součástí požadavků z důvodů flexibility. v době návrhu nejsou plánovány žádné akce, které by do této kategorie měly spadat. Kategorie je připravena pro akce, na které by mělo být spuštěno přihlašování v rámci informačního systému, ale které by nespadaly do žádné z jiných kategorií. Podoba této kategorie je opět blízká víkendovým akcím a táborům. Jedná se tedy přihlašování na jednorázové akce.

3.2 Současný stav

V současné chvíli administrativní pracovníci využívají interní databázi, do které ukládají informace o účastnících akcí a darech. Tato databáze umožňuje vyhledávání v historii, ale vyžaduje vyplnění veškerých údajů zaměstnancem. Samotné přihlašování na akce probíhá obvykle emailem nebo pomocí formulářů Google[4]. Veškeré údaje kontroluje zaměstnanec a manuálně přepisuje do interní databáze a poté do jakýchkoli potřebných seznamů. Databáze evidující dárce je v současnosti přizpůsobená pouze pro osoby, ale dary zasílají i firmy.

Během organizace akce se využívá několik různých druhů seznamů. Před akcí je nutné evidovat přihlášené účastníky včetně případných náhradníků, při příjezdu na akci se obvykle využívá vytištěný seznam určený pro prezenci účastníků, po akci je nutné připravit jiné seznamy pro účetnictví a pro archivaci. Všechny tyto seznamy jsou v současné době udržovány manuálně v tabulkovém procesoru. Před akcí je nutné sledovat například stav zaplacení. Platba může být zaslána elektronicky, přičemž správu účtu má na starosti jiný zaměstnanec než samotnou akci. Druhá možnost je poté platba hotově při příjezdu.

Administrativní pracovníci dále sledují uživatele, kteří jsou nebo měli by být členy SKM[5]. Účastník akcí Domu Ignáce Stuchlého ve Fryštáku se pro daný školní rok stává také členem SKM, ale výjimku z tohoto pravidla mají lidé, kteří se účastní méně než dvou akcí za rok. Pro zaměstnance je tedy důležité sledovat, kteří účastníci jsou přihlášeni za školní rok již aspoň na druhou akci a ještě nejsou pro daný rok členy SKM. Pro takové účastníky je poté připravena přihláška. v současné době kontrola probíhá manuálně postupným vyhledáváním účastníků v interní databázi.

Dalším úkolem administrativního pracovníka je evidence dárců. Opět se jedná o interní databázi, do které jsou vkládány údaje o lidech a firmách, které zašlou organizaci finanční dar. Tyto údaje nejsou veřejné a přístup k nim mají pouze zaměstnanci. v této podobě by evidence měla také zůstat.

DISklub a kroužky v současné chvíli žádnou databázi nevyužívají. Přihlašování probíhá pomocí papírových přihlášek, které v obou případech platí na celý školní rok. Oba typy aktivit jsou ale oddělené. Účastník tak může navštěvovat DISklub a neúčastnit se žádného kroužku nebo naopak docházet do kroužku a nebýt členem DISklubu.

3.3 Požadavky

V následující kapitole vysvětlím požadavky organizace a zaměstnanců na funkce informačního systému. Jedná se o prvky, které v tuto chvíli nemají k dispozici nebo fungují jiným, obvykle složitějším a časově náročnějším, způsobem.

Mělo by se jednat o přihlašovací systém. Ne tedy pouze o evidenci seznamů účastníků, ale o webovou aplikaci, ve které si účastníci sami budou moci vyhledat a prohlížet připravované akce. Na tyto akce se také sami zvládnout přihlásit a vyplnit všechny potřebné údaje. Schvalování přihlášek ale musí zůstat v kompetenci vedoucího a zaměstnanců. Ti budou mít možnost přihlášky přijmout, odmítnout a případně spravovat uživatelé přihlášené po naplnění kapacity jako náhradníky. v případě odhlášení některého z přihlášených účastníků je poté účast nabídnuta účastníkům z tohoto seznamu. u akce by poté měly být účastníkům dostupné důležité dokumenty potřebné pro účast na akci. Jedná se například o potvrzení od lékaře nebo souhlas s pořizováním obrazových záznamů.

[4] https://www.google.com/intl/cs_CZ/forms/about/

[5] Salesiánské kluby mládeže, více viz https://www.skm.cz/

Každý uživatel by měl mít vytvořený účet, pomocí kterého se na akce bude přihlašovat. u uživatelského účtu budou uloženy základní údaje. Při přihlášení vyplní účastník údaje specifické pro danou akci. Mezi tyto údaje může patřit například požadovaná velikost pamětního trička vyráběného pro danou akci. Prostřednictvím svého účtu bude mít uživatel možnost kontrolovat své aktuální přihlášky a případně se z akce odhlásit. Na odhlášení účastníka je vždy upozorněn vedoucí. Po přihlášení na akci je účastníkovi umožněn přístup k seznamu dalších účastníků obsahujícím pouze jména.

Každá akce by měla umožňovat prohlížení a tisk seznamů. Jedná se například o seznamy přihlášených účastníků, náhradníků nebo vedoucích pro prezenci na akci či archivaci. Zároveň by měla být možnost rozeslat hromadný e-mail účastníkům akce.

Některé pořádané akce navazují na jiné. Jedná se například o podzimní setkání účastníků tábora. Na tyto akce by tedy měli být speciálně upozorněni účastníci první akce.

Pro zaměstnance by mělo být možné vyhledávat jak akce konané v minulosti, tak také akce kterých se zúčastnil konkrétní uživatel. Díky tomu bude možné vyhledat například kteří účastníci byli na minulých ročnících stejné akce, jakých akcí se již zúčastnil potenciální vedoucí a také kteří lidé se účastní více než jedné akce za školní rok a měli by se tak stát členy SKM[6]. Zároveň je potřeba pro účely evidence a vykazování činnosti generovat seznamy členů v konkrétním školním roce.

Do evidence dárců je vyžadováno zahrnutí osob a také firem. Tabulka je soukromá pouze pro zaměstnance a pouze zaměstnanci budou mít možnost do ní přidávat údaje.

Zaměstnankyně spravující účetnictví do současné databáze přístup neměla. v informačním systému by bylo vhodné, aby měla sama přístup k evidenci akcí a účastníků a mohla v ní vyhledávat a vytisknout si potřebné seznamy. Zároveň právě ona má přístup k účtu a eviduje platby za akce přijaté přes internet. Právě tyto platby by tak měla mít možnost přímo v systému zapsat a změnit odpovídající údaj u přihlášky.

3.3.1 Personální struktura

Velká část systému bude přístupná jen části uživatelů. Široká veřejnost má možnost prohlížet připravované akce a přihlašovat se na ně, s výjimkou vzdělávacích akcí určených pro vedoucí. Role se v systému mohou prolínat a stejný člověk může jet například zároveň na jednu akci jako vedoucí a na druhou jako účastník. Vedoucí má přístup k více údajům o účastnících akce a získává také přístup k prohlížení a přihlašování na vzdělávací akce.

Možnost organizovat akce mají pouze zaměstnanci Domu Ignáce Stuchlého. Ti zároveň mohou být hlavním vedoucím zodpovědným za danou akci. u akcí vedou evidenci přihlášek a rozhodují o přijetí účastníka. Administrativní pracovník také eviduje dárce a členství účastníků v SKM.

Další rolí je účetní, který přijímá platby zaslané prostřednictvím internetového bankovnictví a po akci vždy dostává seznam účastníků pro vyúčtování.

3.3.2 Uživatelské role

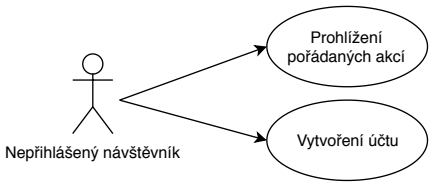
Uživatelé v informačním systému mohou zastávat následující role:

- Běžný uživatel
- Vedoucí
- Ekonom

[6] Salesiánské kluby mládeže. Kdy je členství vyžadováno bylo vysvětleno v kapitole 3.2

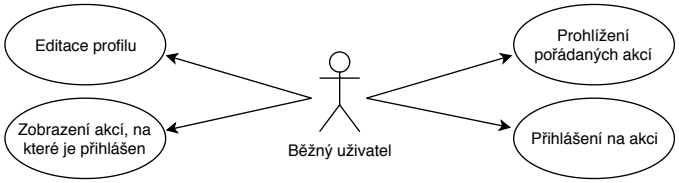
- Zaměstnanec
- Administrátor

Vyšší role vždy dědí všechny pravomoci po předchozí a přidává některé navíc.



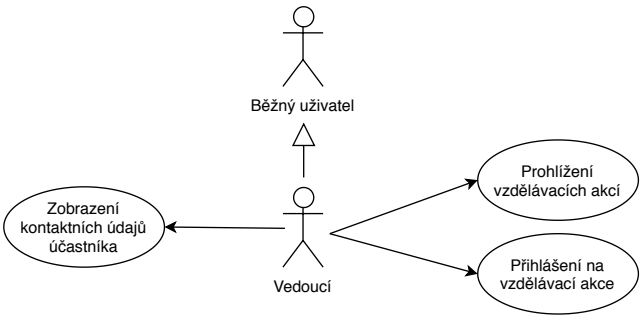
Obrázek 3.1: UC diagram pro uživatelskou roli *nepřihlášený návštěvník*

Nepřihlášený návštěvník může prohlížet připravované akce (s výjimkou vzdělávacích). Pro přihlášení na akci je nutné si nejprve vytvořit uživatelský profil.



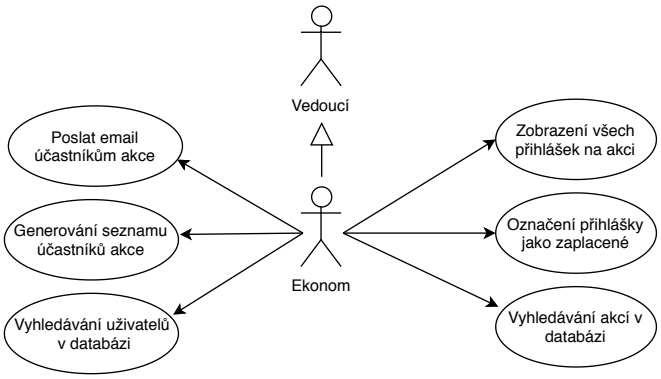
Obrázek 3.2: UC diagram pro uživatelskou roli *běžný uživatel*

Běžný uživatel po přihlášení do aplikace získá možnost se na akce přihlašovat a prohlížet své přihlášky. Po schválení přihlášky na akci vidí také jména ostatních přihlášených účastníků.



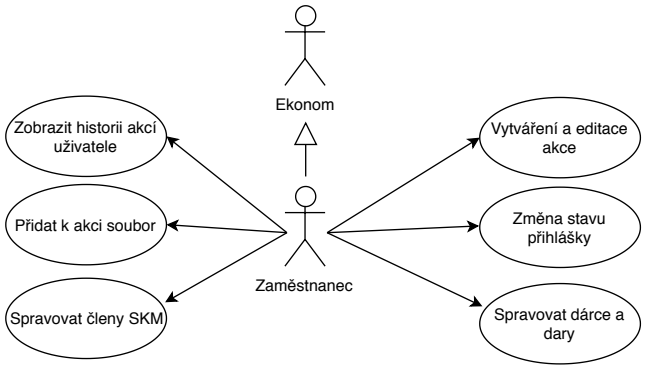
Obrázek 3.3: UC diagram pro uživatelskou roli *vedoucí*

Vedoucí má navíc přístup ke vzdělávacím akcím a u přihlášených účastníků vidí také kontaktní údaje.



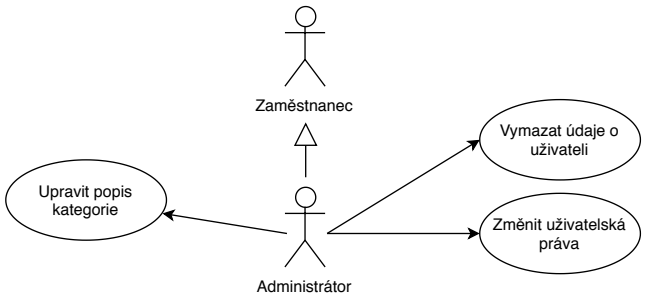
Obrázek 3.4: UC diagram pro uživatelskou roli *ekonom*

Ekonom již má možnost u akce prohlížet přihlášky ve všech stavech a měnit stav zaplacení dané přihlášky. Také má přístup ke generování seznamů účastníků a k vyhledávání uživatelů a akcí v databázi.



Obrázek 3.5: UC diagram pro uživatelskou roli *zaměstnanec*

Zaměstnanec má možnost vytvářet, editovat akce a také může rozhodovat o změně stavu přihlášek. u jakéhokoli uživatele může zobrazit historii akcí. Má možnost spravovat členství v SKM u uživatelů a evidovat dárce.



Obrázek 3.6: UC diagram pro uživatelskou roli *administrator*

Administrátor má právo změnit roli uživatele. Také může měnit popis kategorie akcí a vymazat osobní údaje uživatele z databáze.

3.4 Návrh

V následující kapitole popíši návrh modelu informačního systému. Text navazuje na kapitolu 3.3 a popisuje, jak požadované vlastnosti a realizovat ve výsledném informačním systému.

3.4.1 Databáze

V následující kapitole je popsán návrh databáze a zobrazeny jednotlivé části ER[7] diagramu.

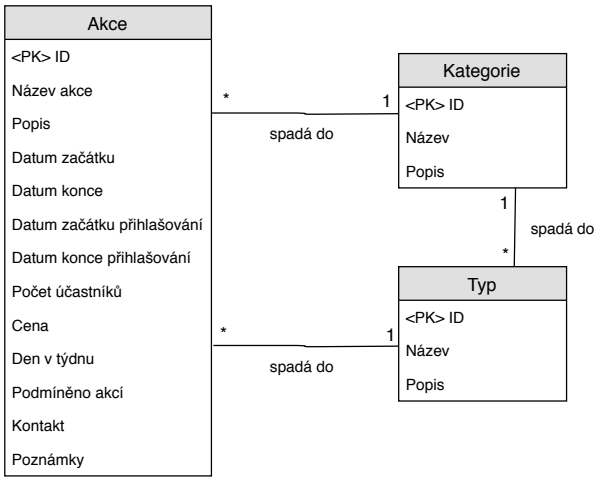
Uživatelé

Obrázek 3.7: ER diagram tabulky `uživatel`

[7]Entity Relationship - diagram znázorňující entity a vztahy mezi nimi

Nejdůležitějšími prvky databáze budou tabulky obsahující uživatele a akce. Vzhledem k faktu, že uživatelské role se mohou v rámci systému prolínat, rozhodl jsem se použít pouze jednu tabulku pro všechny uživatele. Ta obsahuje základní údaje vyžadované u všech nebo aspoň většiny akcí, dále heslo, uživatelská oprávnění a poznámky, které si k uživateli mohou doplnit zaměstnanci a pouze pro ně jsou také viditelné. Doplňkové údaje specifické pro jednotlivé akce budou uloženy ve speciální tabulce.

Akce



Obrázek 3.8: ER diagram tabulek `akce`, `kategorie` a `typ`

Tabulka akcí je opět jedna společná pro akce všech kategorií. Akce obsahuje základní údaje, mezi které patří název, popis, kategorie, typ, datum začátku a konce a cenu. Dále datum kdy bude spuštěno a ukončeno přihlašování. Akce také obsahuje identifikační číslo hlavního vedoucího, který je za akci zodpovědný, a opět poznámky pro zaměstnance. Specifické je ukládání pravidelných akcí. u kroužků je navíc vyplněn den v týdnu, kdy se kroužek koná. Ve sloupcích `datum začátku` a `datum konce` je jako datum uložen začátek a konec školního roku a jako čas počáteční a koncový čas daného kroužku.

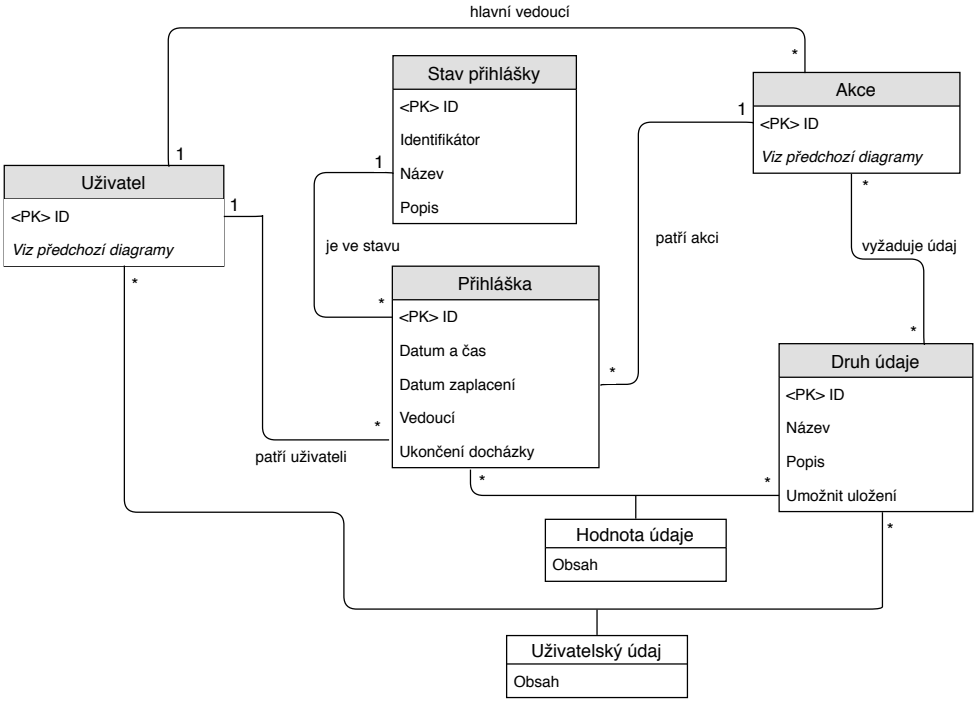
Kategorie a typ mají vlastní tabulky. v těch je definován jejich název a popis. Typ navíc spadá pod určitou kategorii a proto na ni odkazuje.

Tabulka 3.1: Přehled kategorií a typů akcí

Kategorie	Typy, které pod ni spadají
Víkendovky	Střapec-Lis-Košt, duchovní, prázdninové, outdoorové
Tábory	příměstské, pobytové
Kroužky	
DISklub	
Vzdělávací	oDISea, víkendovky
Jiné	

Střapec, Lis a Košt jsou názvy tří pravidelných každoročních akcí pro mládež konaných v Domě Ignáce Stuchlého, oDISea je název vzdělávacího kurzu pro instruktory Orientačních dnů. Kroužky, DISklub a jiné akce žádné další dělení nemají.

Přihlášky



Obrázek 3.9: ER diagram tabulek `přihláška`, `stav přihlášky` a tabulek ukládajících specifické údaje

Informace, které jsou vyžadovány při přihlášení na akci a jsou pro každou akci specifické, ukládá tabulka druhů specifických údajů. Údaje mají svůj název a popis. Na tyto údaje se poté mohou odkazovat jednotlivé akce jako na vyžadované, které poté uživatel musí vyplnit při přihlašovaní na danou akci. Uživatel má také možnost si pro každý druh údaje zadat výchozí hodnotu a při vyplňování dodatečných informací při přihlašování se mu poté ve formuláři tyto hodnoty automaticky objeví. Hodnoty údajů jsou uloženy ve vazebních tabulkách propojujících druh údaje s akcí či uživatelem.

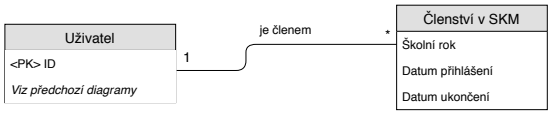
Samotná přihláška má vlastní tabulku. Obsahuje datum a čas svého vytvoření, datum zaplacení a datum ukončení docházky. Odkazuje se na akci a uživatele které propojuje a také na svůj stav. Stavy přihlášky mohou být následující:

- Čekající - uživatel odeslal přihlášku na akci a čeká na schválení vedoucím
- Přihlášen - přihláška byla potvrzena zaměstnancem
- Náhradník - přihláška byla přijata, ale uživatel je označen jako náhradník
- Žádost o odhlášení - uživatel vyjádřil zájem se z akce odhlásit

- Odhlášen - odhlášení bylo potvrzeno zaměstnancem
- Zrušeno - uživatel zrušil přihlašování ještě před schválením ze strany zaměstnance
- Zamítnuto - zaměstnanec zamítl přihlášku
- Vedoucí - uživatel je na akci přihlášen v roli vedoucího

Při přihlašování i odhlašování z akce přihláška vždy čeká na potvrzení zaměstnancem. Ten tak má při přihlašování právo rozhodnout, zda bude uživatel na akci přijat a případně jej označit jako náhradníka. Při odhlašování se opět vyžaduje potvrzení žádosti zaměstnancem. Ten má zároveň možnost vybrat případného náhradníka, který bude na akci pozván místo odhlášeného uživatele.

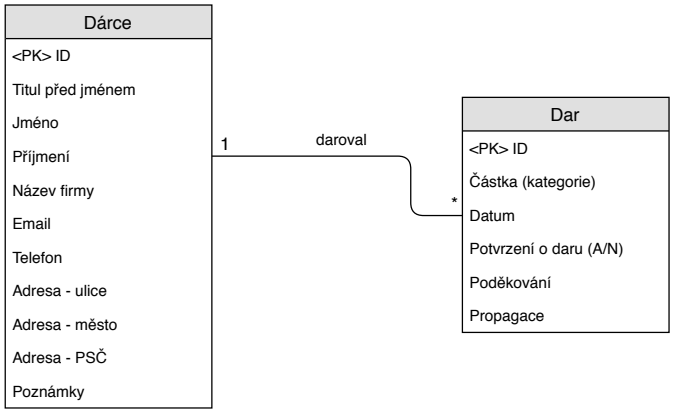
Členství v SKM



Obrázek 3.10: ER diagram tabulky `člen SKM`

Další tabulka eviduje členství v SKM[8] u jednotlivých uživatelů. Členství může být na jeden školní rok, nebo na dobu neurčitou. Tabulka obsahuje sloupce pro školní rok, datum přihlášení a datum ukončení členství. Školní rok je reprezentován rokem, ve kterém daný školní rok začíná. Členství na dobu neurčitou je reprezentováno hodnotou `9999`.

Dárci

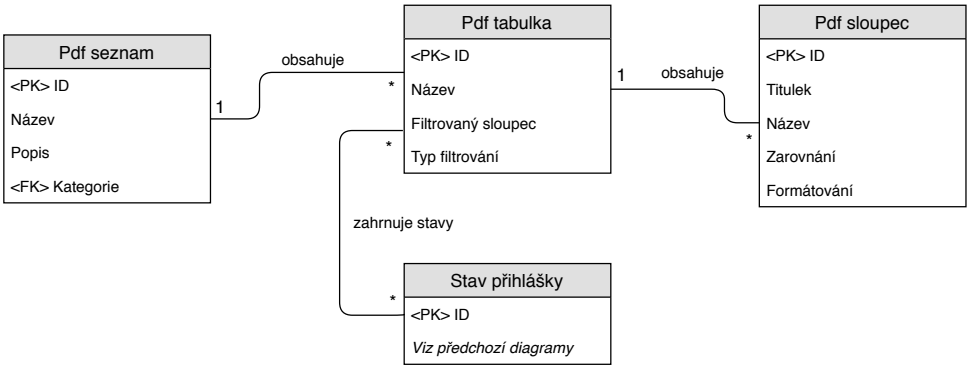


Obrázek 3.11: ER diagram tabulek `dárce` a `dar`

[8] Salesiánské kluby mládeže. Kdy je členství vyžadováno bylo vysvětleno v kapitole 3.2

Správu dárců zajišťují dvě tabulky, které jsou oddělené od všech ostatních. První z nich obsahuje údaje o dárcích. Pokud je dárcem osoba, sloupec `název firmy` zůstane nevyplněný. v opačném případě je `název firmy` vyplněn a uvedená osoba vystupuje v roli kontaktní osoby za danou firmu. Druhá tabulka obsahuje záznamy o jednotlivých darech.

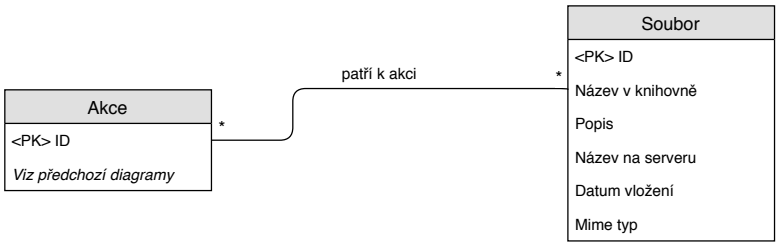
Generování seznamů



Obrázek 3.12: ER diagram tabulek využívaných pro uložení struktury generovaných seznamů

Při evidenci účastníků je nutné generovat velké množství různých seznamů. Následující tabulky obsahují záznamy popisující obsah a vzhled jednotlivých seznamů. Konkrétní seznam má v tabulce `Pdf seznam` definovaný název a popis pro uživatele. Jeden seznam může obsahovat větší množství tabulek. Ty jsou určeny v tabulce `Pdf tabulka` obsahující titulek tabulky. k této tabulce se váže libovolné množství stavů přihlášek. v tabulce jsou poté zobrazeni pouze uživatelé, kteří na danou akci mají vytvořenou přihlášku v jednom ze zvolených stavů. Každá tabulka seznamu má poté v tabulce `Pdf sloupce` definovány položky, které se mají u každého uživatele zobrazit. Sloupec má v databázi definovaný titulek a název dat, které se mají ve sloupci zobrazit. Dále je možné specifikovat zarovnání textu ve sloupci a formátování, které spočívá v obarvení dat ve sloupci podle pohlaví uživatele nebo podle stavu přihlášky.

Knihovna souborů



Obrázek 3.13: ER diagram tabulky `soubor`

Evidenci souborů dostupných u jednotlivých akcí zajišťuje tabulka `soubor`. Ta obsahuje pro každý soubor popis a dva názvy. Jeden odpovídá názvu souboru uloženého na serveru a druhý určuje, jak bude soubor reprezentován v rámci přihlašovacího systému. Další uložené údaje jsou typ souboru formátovaný podle standardu MIME[9] a datum vložení souboru do systému.

[9]Multipurpose Internet Mail Extensions, seznam registrovaných typů souborů je dostupný na adrese http://www.iana.org/assignments/media-types/media-types.xhtml

Kapitola 4

Využité technologie

Obsahem této kapitoly je popis technologií využitých při tvorbě informačního systému v rámci této bakalářské práce. Podkapitoly jsou seřazeny od databáze přes jazyk PHP a framework Nette až po jazyky použité pro návrh webové stránky jako uživatelského rozhraní.

4.1 SQL

SQL[1] je prvním, nejrozšířenějším a také jediným standardizovaným široce přijímaným databázovým jazykem. Jedná se o neprocedurální jazyk, což znamená, že uživatel zadává pouze jaká data potřebuje a nemusí řešit jak se získávají. Jazyk je založen na anglických slovech používaných v příkazech. Je tedy poměrně jednoduché se základní příkazy naučit.[3]

4.2 MariaDB

Jedná se o systém řízení báze dat, který je odvozen ze systému MySQL[2]. MySQL byl jeden z nejrozšířenějších systémů. Vyvíjen byl původně s otevřeným zdrojovým kódem. i přesto ale zdarma poskytoval srovnatelné možnosti s komerčními alternativami společností Microsoft a Oracle. Jeho silnými stránkami jsou vysoký výkon, přenositelnost mezi platformami a spolehlivost.[19]

V roce 2009 systém MySQL přešel pod správu společnosti Oracle. Michael Widenius, původní tvůrce MySQL, na tuto změnu okamžitě reagoval vytvořením nové vývojové větve. Veden byl svými nesympatiemi ke společnosti Oracle a obavou o další vývoj systému MySQL. Byl tak vytvořen nový systém nazvaný MariaDB[3]. Ten má za cíl nabídnout minimálně stejné funkce jako MySQL, po určitou dobu udržovat kompatibilitu a zachovat nový systém jako svobodný software.[11]

4.3 PHP

Původním významem zkratky PHP bylo *Personal Home Page*. Význam této zkratky (v překladu *osobní domovská stránka*) vystihuje hlavní úkol, který PHP dodnes má. Tím je vytvoření webové stránky nebo částí stránky dynamicky na základě například přihlášeného

[1] Structured Query Language - strukturovaný dotazovací jazyk
[2] https://www.mysql.com/
[3] https://mariadb.org/

uživatele nebo aktuálních dat z databáze. Postupem času rostlo rozšíření této technologie a také její možnosti. Mimo jiné se změnil význam zkratky na aktuálně používané *PHP: Hypertext Preprocessor*. PHP je skriptovacím jazykem, který lze vložit přímo do HTML kódu. Webový server poté před odesláním této stránky spustí PHP skript a klientovi odešle výsledek ve formě HTML.[19]

Mezi výhody PHP patří například dostupnost zdarma s open source licencí. Jedná se o velmi rozšířený jazyk nezávislý na konkrétním operačním systému. Lze také jednoduše využít jen část jeho potenciálu, což snižuje náročnost poznávání systému pro nové programátory. Díky svému zaměření také PHP obsahuje mnoho funkcí používaných při tvorbě webových stránek. Jedná se například o usnadnění práce s databází, správu cookies a formátování HTML kódu.[17]

Frameworky

Pojem *framework* lze vystihnout několika způsoby. Jedním z nich je například tento: „*Aplikační framework je sada funkcí, tříd a konvencí, které usnadňují řešení běžných úloh*“[17]. Pro vývoj webových aplikací v PHP se využívá velké množství různých frameworků. Informační systém tvořený v této bakalářské práci využívá Nette framework, kterému se věnuje kapitola 4.4. Další rozšířené frameworky jsou například Laravel[4], Symfony[5] a Zend[6]

4.4 Nette

Nette[7] je český framework, který má za cíl usnadnit programátorům vyhnout se opakování stejného kódu a mít výsledný kód přehledný. Nabízí například šablonovací systém, ladící nástroje a databázovou vrstvu. Důraz klade na bezpečnost a velkému množství bezpečnostních rizik automaticky předchází. Framework se také snaží vést programátory k psaní čistého a udržitelného kódu.[16]

4.4.1 Architektura Model-View-Controller

Architektura nazývaná Model-View-Controller je třívrstvá architektura oddělující databázovou vrstvu, aplikační vrstvu a uživatelské rozhraní. Tyto vrstvy spolu navzájem komunikují, ale každá může být při zachování stejného komunikačního rozhraní vyměněna nezávisle na ostatních. Toto rozdělení koresponduje s architekturou zobrazenou v kapitole 2.3.1.

Model

Jedná se o vrstvu aplikace komunikující s databází. Jejím úkolem je reagovat na požadavky aplikační vrstvy a odeslat jí zpět informace získané z databáze. Ve frameworku Nette roli modelu zastávají třídy nazvané *manager* s prefixem odpovídajícím spravované oblasti. Například tedy *UserManager* jako model spravující uživatelskou část databáze. Tato třída obsahuje funkce volané z aplikační vrstvy. v konkrétní funkci je poté proveden databázový dotaz a jeho výsledek je její návratovou hodnotou.

[4] https://laravel.com/
[5] https://symfony.com/
[6] https://framework.zend.com/
[7] https://nette.org/

Controller

Controller je aplikační vrstva, která zajišťuje samotný běh aplikace a propojuje databázovou vrstvu s uživatelským rozhraním. Jejím úkolem je vyžádat si od databázové vrstvy informace z databáze, které poté zpracuje a připraví ke zobrazení v uživatelském rozhraní. Nette framework používá místo slova *Controller* pojem *Presenter*. Název třídy i souboru je vytvořen stejným způsobem jako u modelu. v případě presenteru pracujícím s uživateli to bude například *UserPresenter*. Tato třída obsahuje funkce odpovídající výsledným webovým stránkám aplikace generovaným v PHP a funkce reagující na požadavky zaslané prostřednictvím technologie AJAX (viz kapitola 4.9). v prvním případě funkce připraví všechna data potřebná pro vykreslení stránky a tu poté framework vykreslí podle připravené šablony (viz kapitola 4.4.2). Při ajaxovém požadavku se obvykle jedná jen o malou část dat, kterou okamžitě po získání odesílá. v obou případech jsou data obvykle získána pomocí volání funkce obsažené v modelu.

View

Vrstva uživatelského rozhraní je výsledná webová stránka zobrazená u uživatele. Využívá jazyky HTML, CSS a Javascript popsané v následujících kapitolách. v Nette frameworku je sestavena na základě dat poskytnutých aplikační vrstvou podle odpovídající šablony (viz kapitola 4.4.2). Stránka může obsahovat také kód v jazyce Javascript a získávat data asynchronně prostřednictvím technologie AJAX (viz kapitola 4.9).

4.4.2 Latte

Latte je šablonovací systém využívaný v Nette frameworku. Důraz klade na rychlost výsledného PHP kódu, bezpečnost a srozumitelnost pro programátory[9]. Samotná šablona může obsahovat pouze obyčejný HTML dokument. Navíc ale nabízí možnost využití *maker*. Tyto makra mohou být buď uzavřeny do složených závorek a nabo jsou vloženy do HTML tagu s prefixem `n:`. Makra umožňují rozšířit možnosti jazyka HTML například o cykly a podmínky. Dále je do složených závorek možno psát PHP kód pracující s proměnnými, které do šablony vložil presenter.

V rámci maker lze také jednoduše generovat odkazy na jiné stránky v rámci aplikace. v latte šabloně stačí uvést cílový presenter a view. Nette se při zpracování postará o vytvoření odkazu a kontrolu, zda odkazovaná stránka v aplikaci existuje.

4.4.3 Database Explorer

Jedná se knihovnu usnadňující přístup k databázi zahrnutou do Nette frameworku. Knihovna nevyžaduje od programátora psaní SQL dotazů. Programátor pracuje s objekty jazyka PHP. Základním objektem je databázový kontext, nad kterým je volána metoda `table`. Nad výsledkem lze poté řetězit další funkce, které umožňují provádění databázových operací jako jsou výběr, vložení nebo odstranění řádků. Database Explorer se poté na jejich základě podle dat které jsou na stránce skutečně využity snaží vytvořit co nejefektivnější dotaz a ten použít.[4]

4.4.4 mPDF

Pro generování seznamů ve formátu PDF jsem využil knihovnu *mPDF*[8], která není standartní součástí frameworku Nette. Ta dokáže převést HTML kód do dokumentu ve formátu PDF. Pro generování vstupního HTML kódu využívám Latte šablony popsané v kapitole 4.4.2.

4.5 HTML

Jazyk HTML je jedním ze základních pilířů webu. Slouží k popisu toho, co vlastně stránka obsahuje. Zkratka názvu jazyka znamená *Hypertext Markup Language*, tedy *Hypertextový značkovací jazyk*. Slovo *hypertextový* naznačuje možnost odkazů jednotlivých stránek na ostatní. *Značkovací jazyky* obecně doplňují text o tzv. *značky*, neboli *tagy*, které popisují význam daného textu. Každá značka dále může obsahovat atributy, které dále upřesňují její parametry. Značky jsou nejčastěji párové a jejich obsahem je text uzavřený mezi počáteční a koncovou značku. Nepárovými jsou například značky `img` používané pro vkládání obrázků nebo `br` označující zalomení řádků. Jednotlivé značky do sebe lze zanořovat a vytvářejí tak hierarchickou strukturu dokumentu.[6]

4.6 CSS

Zkratka CSS[9] označuje jazyk určený pro popis kaskádových stylů HTML dokumentu. Samotný jazyk HTML je určený především pro popis obsahu. Pomocí kaskádových stylů lze poté definovat vzhled výsledné stránky.[6]

Kód jazyka CSS se skládá ze selektoru, určujícího na jaké elementy HTML dokumetnu se vztahuje. Selektor může využívat názvy značek nebo také atributy `id` a `class` přiřazující elementu jednoznačný identifikátor nebo třídu. Za selektorem následuje popis vlastností a jejich hodnot. Takovou vlastností může být například barva textu. Hodnota, která je jí přiřazena, je poté aplikována na všechny zvolené elementy.

4.7 jQuery

Jedná se o knihovnu k programovacímu jazyku Javascript. Ten je, na rozdíl od PHP, interpretován až v internetovém prohlížeči na počítači klienta a využívá se tak na programování interaktivních prvků[17].

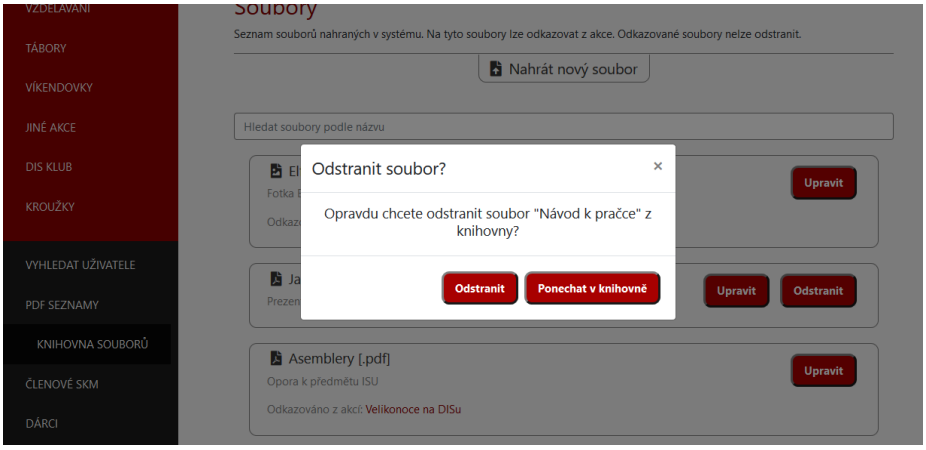
Implementace jazyka Javascript se v různých prohlížečích liší. Knihovna jQuery má proto za cíl tyto rozdíly odstranit. Kód napsaný pomocí knihovny jQuery by měl ve všech prohlížečích fungovat stejně. Navíc tato knihovna umožňuje řetězení volání metod a tím kratší zápis kódu.[1]

4.8 Bootstrap

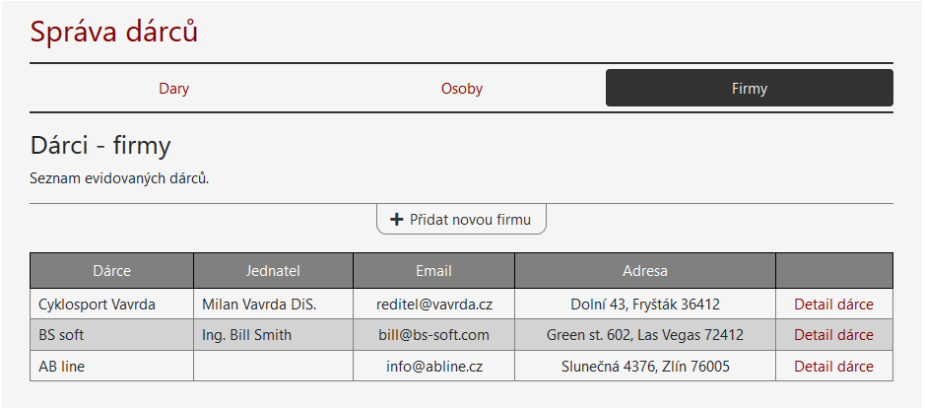
Bootstrap je knihovna obsahující rozšíření pro HTML, CSS i Javascript. Je určená pro vývoj interaktivního uživatelského rozhraní webové stránky. Obsahuje velké množství komponent, které lze v projektu využít a usnadňuje návrh responzivních stránek.[2]

[8] https://mpdf.github.io/
[9] Cascading Style Sheets

Ve své aplikaci využívám *Grid system* pro tvorbu rozložení stránek, které se dynamicky mění při změně velikosti okna. Dále komponenty *Modal*, vytvářející dialogová okna, a *Pills* pro přepínání mezi více druhy obsahu na stejné stránce.



Obrázek 4.1: Dialogové okno vytvořené pomocí Bootstrap komponenty *Modal*



Obrázek 4.2: Záložky *Dary*, *Osoby* a *Firmy* vytvořené pomocí Bootstrap komponenty *Pills*

4.9 AJAX

Technologie AJAX[10] umožňuje načítat a překreslovat jednotlivé části webové stránky samostatně. Cílem je zabránit opakovanému posílání stejných dat mezi serverem a klientem. Protokol HTML je bezstavový, což v praxi znamená, že jakékoli stavové informace je nutné posílat při každém požadavku na server a opačně. Zkratka je složena z názvů technologií, ale pojem AJAX je využíván obecněji a nevyžaduje použití Javascriptu, XML ani asynchronního přístupu. Poznávacím znamením tak zůstává odeslání požadavku na pozadí a aktualizace části webové stránky po přijetí odpovědi.[8]

V této aplikaci je AJAX využíván nejčastěji při vyhledávání akcí či uživatelů podle jména.

[10] Asynchronous Javascript and XML

Kapitola 5

Implementace

V této kapitole popíši strukturu výsledné aplikace a implementaci jednotlivých prvků. Základní struktura a umístění souborů vychází z doporučené struktury používané Nette frameworkem. Ta je podle [10] následující:

```
sandbox/
├── app/                          adresář s aplikací
│   ├── config/                   konfigurační soubory
│   │   ├── config.neon           konfigurační soubor
│   │   └── config.local.neon
│   │
│   ├── forms/                    třídy formulářů
│   ├── model/                    modelová vrstva a její třídy
│   ├── presenters/               třídy presenterů
│   │   └── templates/            adresář se šablonami
│   │       └── @layout.latte     šablona společného layoutu
│   ├── router/                   třídy routerů
│   └── bootstrap.php             zaváděcí soubor aplikace
│
├── log/                          obsahuje logy, error logy atd.
├── temp/                              pro dočasné soubory, cache, ...
│
├── vendor/                       adresář na knihovny (např. třetích stran)
│   ├── nette/                    všechny knihovny Nette Frameworku
│   │   └── nette/Nette           framework nainstalovaný Composerem
│   ├── ...
│   └── autoload.php              soubor načítající nainstalované balíčky
│
└── www/                          veřejný adresář, document root projektu
    ├── .htaccess                 pravidla pro mod_rewrite
    ├── index.php                 který spouští aplikaci
    └── images/                   další adresáře, třeba pro obrázky
```

V této kapitole jsou popsány především soubory vytvořené při programování popisovaného informačního systému. Následující struktura zobrazuje adresáře, ve kterých se tyto soubory nachází. Zbývající adresáře jsou popsány jen okrajově nebo nejsou zmíněny vůbec.

```
xmechl00-BP/
├── app/                          adresář s aplikací
│   ├── components/               komponenty využívané v aplikaci
│   ├── model/                    modelová vrstva a její třídy
│   └── presenters/               třídy presenterů
│       └── templates/            složky obsahující šablony presenterů
├── files/
│   └── event_files/              ukládání souborů přiložených k akcím
└── www/
    ├── images/                   obrázky
    ├── css/                      kaskádové styly
    └── js/                       soubory v jazyce Javascript
```

5.1 Modely

Jedná se o část aplikace odpovídající databázové vrstvě. Jak již bylo popsáno v kapitole 4.4.1, název objektu v jazyce PHP se podle konvence využívané v Nette frameworku jmenuje názvem odpovídajícím činnosti modelu s příponou `Manager`. Hlavními modely v aplikaci jsou `AccountManager`, `EventManager` a `ApplicationManager`. Tyto modely provádějí databázové dotazy týkající se uživatelů, akcí a přihlášek. Další modely se týkají generování seznamů, ukládání souborů, odesílání emailů, správy členů SKM, evidence dárců a ve složce s modely je vložený také autentikátor popsaný podrobněji v kapitole 5.4.1.

AccountManager

Tento model má na starosti databázové dotazy týkající se uživatelů. Jeho nejpoužívanější metoda je `getUser`, která na základě identifikačního čísla najde informace o konkrétním uživateli. Další skupinou jsou metody upravující uživatelské účty. Jedná se o registraci nového uživatele, editaci údajů o stávajícím a změnu hesla. Další funkce umožňují vyhledávání uživatelů v databázi. Funkce `searchUserByName` vyhledává pouze podle jednoho parametru, který je porovnáván s textem jména i příjmení spojených do jednoho řetězce. Druhá metoda je `searchUsers`, která přijímá jako samostatné parametry jméno, příjmení, email, uživatelská práva a dva dny, mezi kterými se má nacházet datum narození. Funkce poté zjistí, které parametry jí byly předány vyhledá uživatele splňující všechna zadaná kritéria.

EventManager

Model spravující akce má, obdobně jako `AccountPresenter`, nejpoužívanější metodu nazvanou `getEvent`, která vyhledává konkrétní akci podle zadaného identifikačního čísla. Zbývající metody lze rozdělit do několika typů.

První skupinou jsou funkce vracející seznam akcí. Jedná se o seznam akcí spadajících do konkrétní kategorie či typu. Rozdíl je v přijímaném parametru. Vyhledávání kategorie probíhá na základě textového identifikátoru a vyhledávání typu na základě identifikačního čísla. Rozdíl je daný způsobem, jakým jsou metody volány z presenteru. Požadovaná kategorie je v odpovídajícím presenteru zadávána programátorem, zatímco typy spadající pod kategorii jsou vyhledány programem a ten poté do volání vloží odpovídající identifikační číslo. Kategorie i typy jsou načítány z databáze v metodě `getPossibleCategories`. Její

návratovou hodnotou je asociativní pole obsahující údaje o kategorii i pod ni spadajících typech.

Další skupinou jsou metody získávající údaje spojené s konkrétní akcí. Jedná se o seznamy přihlášek či vedoucích získaných na základě identifikačního čísla akce.

Metody `createNewEvent` a `updateEventInfo` obstarávají vytváření nové akce a editaci již existující.

Zbývající metody umožňují vyhledávání v tabulce akcí. Jedná se o metody vyhledávající na základě textu obsaženého v názvu akce a povoleného rozsahu dnů, ve kterých se vyhledaná akce má nacházet. Omezení je možné na základě kategorie či typu.

ApplicationManager

Tento model spravuje přihlášky a obsahuje největší počet metod. Obsahuje metody pro vytváření a mazání přihlášek. Na základě identifikačního čísla uživatele lze vyhledat akce na které je přihlášený jako účastník či vedoucí. Podle identifikačního čísla akce lze vyhledat seznam všech účastníků či účastníků s konkrétním stavem přihlášky.

Další metody umožňují změnu stavu přihlášky nebo nastavení příznaku zaplacení. Pomocí metody `getUserToEventApplication` vyhledat přihlášku konkrétního uživatele na konkrétní akci a získat informace o jeho přihlášce či hodnotu `null` pokud uživatel na akci přihlášený není.

Zbývající metody zajišťují uložení či načtení specifických informací vyžadovaných danou akcí.

ListManager

`ListManager` zajišťuje správu uložených předpisů pro generování seznamů. Obsahuje metody pro načtení, vytvoření a úpravu dat v tabulkách `pdf_seznam`, `pdf_tabulka`, `pdf_sloupec` a `pdf_tabulka_stavy`.

Speciální metodou je `getEventApplicationList`. Tato metoda má jako návratovou hodnotu asociativní pole vytvořené speciálně pro generování seznamu a obsahující údaje o uživatelích i jejich přihláškách.

FileManager

Tento model spravuje údaje o souborech uložených v knihovně přihlašovacího systému. Samotné uložení souborů na server je zpracováno ve třídě `FilePresenter`.

Základní metody obsažené v této třídě umožňují uložení údajů o souboru, jejich úpravu a odstranění. Další metody umožňují získat údaje o souboru na základě identifikačního čísla, seznam všech souborů, seznam souborů připojených ke konkrétní akci nebo vyhledávat soubory podle názvu.

Zbývající metody zajišťují připojení souboru k akci ve vazební tabulce `soubor_akce` či naopak odstranění tohoto propojení.

MemberManager

Model spravující členství v SKM umožňuje výpis z tabulky `clen_skm` na základě identifikačního čísla uživatele nebo školního roku. Mezi členy s aktivním členstvím jsou zahrnuti i uživatelé s členstvím na dobu neurčitou aktivním v daném školním roce. Funkce

`getPossibleYears` má za úkol vyhledat nejstarší záznam a jako náratovou hodnotu vytvoří asociativní pole obsahující školní roky od nejstaršího záznamu do současnosti. Hodnoty v poli jsou poté využity jako položky pro výběr zobrazeného školního roku u seznamu členů.

Metody `addMembership`, `cancelMembership` a `removeMembership` zajišťují správu evidovaných členství. Odstranění pomocí `removeMembership` je využíváno pro vymazání záznamu z databáze například při chybném vytvoření záznamu. Metoda `cancelMembership` členství ponechá v databázi, ale nastaví pro něj datum ukončení.

Důležitou metodou je `getPotentialMembers`. Tato metoda vyhledá uživatele, kteří v zadaném období nebyli členy a zúčastnili se minimálně dvou akcí. Návratovou hodnotou je vícerozměrné asociativní pole obsahující údaje o uživateli a pro každého uživatele seznam akcí, na které byl v zadaném období přihlášen.

DonorManager

`DonorManager` zajišťuje evidenci dárců a darů. Metody pro získání darů umožňují vybrat konkrétní dar podle identifikačního čísla, dary podle konkrétního dárce, podle zadaného období nebo všechny dary v databázi.

Další skupina metod vrací dárce nebo seznam dárců. Děje se tak na základě identifikačního čísla dárce nebo identifikačního čísla daru. Také lze načíst všechny osoby či společnosti uložené v databázi.

Ostatní metody umožňují vytváření a editaci dárců i darů.

MailManager

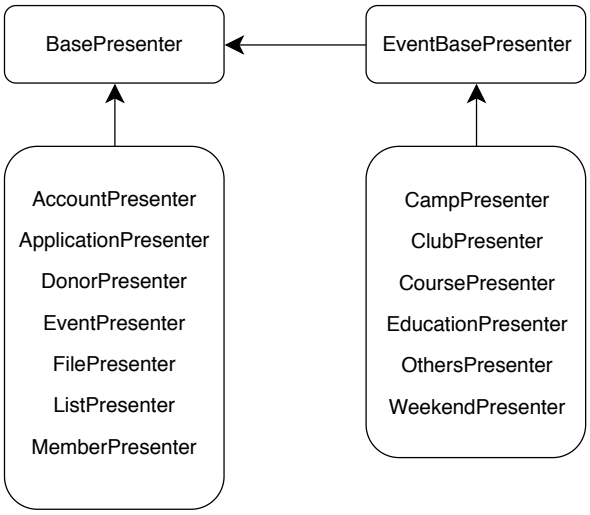
Poslední model se od ostatních liší tím, že nepřistupuje do databáze. Jeho funkce je odeslání emailu. Pro jeho vytvoření je využita Latte šablona, kterou naplní potřebnými údaji. Následně vytvoří instanci třídy `Message` a odešle ji pomocí třídy `SmtpMailer`. Obě třídy jsou součásti frameworku Nette.

Podporované typy emailů jsou dva. Prvním je email odesílaný účastníkům akce a druhým zaslání odkazu pro obnovení hesla.

5.2 Presentery a pohledy

Role controlleru v MVC architektuře byla popsána v kapitole 4.4.1. Presenter je v Nette frameworku obdobou controlleru. Má za úkol zpracování požadavků uživatele, na jejichž základě volá metody poskytované modelem a připravuje data, která poté vykreslí pohled.[10]

Presentery i pohledy jsou navzájem provázané a pohled vždy spadá pod konkrétní presenter. Proto jsem se rozhodl je popsat v jedné kapitole rozdělené podle oblastí, které v aplikaci pokrývají. Třídy reprezentující presentery v jazyce PHP mohou od sebe navzájem dědit a vytvářet tak hierarchickou strukturu. Tato struktura je pro popisovanou aplikaci znázorněna na obrázku 5.1. Názvy jsou, obdobně jako u modelů, tvořeny slovem odpovídajícím činnosti presenteru s příponou `Presenter`.



Obrázek 5.1: Struktura dědičnosti presenterů použitých v aplikaci

5.2.1 BasePresenter

Základní presenter, od kterého dědí všechny ostatní. Ve svém konstruktoru obsahuje inicializaci jednotlivých modelů. Zároveň provadí definici globálních konstant. Ty jsou buď stálé a mohou být upraveny pouze programátorem ve zdrojovém kódu `BasePresenteru`, nebo se jedná o často využívaná data uložená v databázi. Do první skupiny patří například:

- asociativní pole mapující zkratku dne v týdnu na celý český název (například `"po" => "Pondělí"`)
- algoritmus využívaný pro vytváření otisků hesel a jeho cena
- měsíc a den ve kterém se mění školní rok považovaný za aktuální

Druhou skupinu reprezentují například možné stavy, ve kterých se může vyskytovat přihláška. Tyto stavy jsou uložené v databázi. `BasePresenter` proto načte tyto data a po zpracování z nich vytvoří globálně přístupnou konstantu. v tomto případě se jedná o vícerozměrné asociativní pole obsahující všechny informace z tabulky. Pod klíčem odpovídajícím identifikačnímu číslu stavu je uloženo další asociativní pole s údaji o daném stavu.

Stav obsahuje ID, název, popis a identifikátor. Tento identifikátor je krátké slovo bez diakritiky unikátní v dané tabulce, které může být použito jako klíč v asociativním poli. Pro zjednodušení práce v modelu je definováno také jiné asociativní pole mapující identifikátory na ID. Jeho položky tedy mají jako klíč řetězec obsahující identifikátor a hodnotou je identifikační číslo daného stavu (například `"tabory" => 2`).

Dalším asociativním polem definovaným při vytváření instance třídy `BasePresenter` je mapování identifikačních čísel kategorií akcí na odpovídající presentery a pohledy. Toto pole využívá možných kategorií načtených z databáze ve kterých se orientuje pomocí jednoznačných textových identifikátorů. Pomocí identifikátoru `"tabory"` například pozná, že se jedná o kategorii *Tábory* a programátorem je ve třídě `BasePresenter` definováno, že táborům odpovídá presenter `Camp` a pohledy `CampDetail` a `CampManagement`. Pohledy jsou

vždy uloženy dva, jeden pro běžné uživatele a druhý pro uživatele s oprávněním ke správě akcí.

5.2.2 EventBasePresenter

Presenter nazvaný *EventBase* je přechodem mezi `BasePresenter` a presentery obsluhujícími jednotlivé typy akcí. Obsahuje především metody pro zpracování signálů společné pro všechny akce. Těmi jsou například vytvoření přihlášky uživatele na akci a metody pro manipulaci s vybraným uživatelem.

U pohledu zobrazujícího konkrétní akci lze vybrat konkrétního uživatele. Po výběru se zobrazí základní informace o uživateli. Ty jsou doplněny o možnosti přihlášení uživatele na akci nebo naopak úprava jeho přihlášky či odhlášení pokud již přihlášený je. Podrobnější popis se nachází v kapitole 5.2.5.

5.2.3 Uživatelé

Správu uživatelů obstarává `AccountPresenter`. Jedná se především o registraci nového uživatele nebo úpravu údajů. Odhlášení bylo přesunuto jako reakce na signál do třídy `BasePresenter`. Výhodou tohoto řešení je, že metoda pro odhlášení je přístupná z jakéhokoli pohledu. Lze tak jednoduše uživatele odhlásit a zůstat na stejné stránce.

Pod `AccountPresenter` spadá také pohled umožňující zaměstnancům vyhledávat v databázi uživatelů a zobrazení údajů o uživatelském účtu. Zobrazení detailů o účtu, který nepatří přihlášenému uživateli je možné pouze s patřičným oprávněním.

5.2.4 Přihlášky

Tento presenter obsahuje metodu používané pro změny týkající se přihlášek. Jedná se například o změnu stavu přihlášky nebo nastavení dne jejího zaplacení. Samotné vytvoření přihlášky je řešeno formou zpracování signálu. Tato metoda je z důvodu snadnějšího přesměrování v rámci aplikace přesunuto do presenteru `EventBase` od kterého jej poté dědí presentery určené pro konkrétní typ akcí.

Zajímavou částí, která spadá pod `ApplicationPresenter`, je vyplnění specifických informací vyžadovaných konkrétní akcí. Základní uživatelské údaje jsou uloženy v databázi v tabulce uživatelů, ale každá akce může vyžadovat pro přihlášení doplňující specifické údaje. Mezi ty může patřit například dieta, zdravotní omezení, nebo požadovaná velikost památečního trička. Tyto údaje jsou ukládány pro každou akci zvlášť a jeden uživatel tak má možnost zadat pro různé akce různé doplňující údaje.

Pokud se uživatel přihlašuje na akci, která tyto údaje nevyžaduje, je okamžitě vyvolán signál `MakeApplication`, který je zpracován v `EventBasePresenteru`. v případě vyžadovaných specifických informací, je uživatel přesměrován na pohled `enterSpecificInfo` spadající pod `ApplicationPresenter`. Při vytváření této stránky je z databáze načten seznam požadovaných údajů a na jejich základě je vygenerován specifický formulář obsahující údaje požadované danou akcí. Po vyplnění a odeslání formuláře je vytvořena přihláška, do databáze jsou uloženy další potřebné údaje a uživatel je přesměrován zpět na stránku s detaily akce.

5.2.5 Presentery pro jednotlivé typy akcí

Jedná se o skupinu presenterů dědící vlastnosti od preseteru `EventBasePresenter`. Konkrétně se jedná o následující:

- `CampPresenter`
- `WeekendPresenter`
- `CoursePresenter`
- `ClubPresenter`
- `EducationPresenter`
- `OthersPresenter`

Tyto presentery mají jako hlavní úkol zobrazení akce daného typu a také její vytvoření a editaci. Zvažoval jsem vytvoření jednotného presenteru pro všechny typy akcí, ale specifika práce s jednotlivými typy akcí byly nakonec důvodem pro vytvoření presenteru `EventBase`, obsahujícím všechny části které lze u rozdílných akcí sdílet, a specifických presenterů pro každou kategorii, které umožní přizpůsobení požadavků týkajících se pouze jedné kategorie.

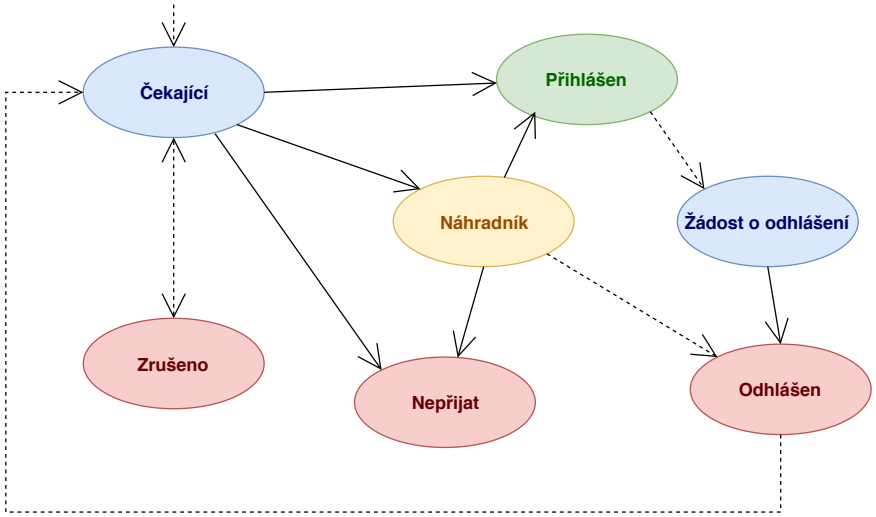
Pohled `*Category*Detail` (za předponu `*Category*` je vždy dosazeno slovo odpovídající kategorii z množiny Camp, Weekend, Course, Club, Others) je přístupný všem uživatelům systému i bez přihlášení. Všem návštěvníkům se zobrazí údaje o akci. Uživatel přihlášený do systému má možnost se na akci přihlásit. Uživateli přihlášenému na akci se zobrazí také jmenný seznam přijatých účastníků a podrobnosti o jeho vlastní přihlášce. Do těch patří stav přihlášky, stav zaplacení a zadané specifické údaje. Uživatel má také možnost požádat o odhlášení z akce, požádat o opětovné přihlášení, nebo zrušit svou ještě nevyřízenou žádost. Opětovné přihlášení je možné jen v případě, že uživatel byl odhlášen nebo sám svou předchozí žádost o přihlášení zrušil. Pokud byla jeho přihláška zamítnuta zaměstnancem, je k jeho opětovnému přihlášení oprávněn pouze zaměstnanec. Výjimkou je pohled `EducationDetail`, který je přístupný pouze vedoucím.

Pro zaměstnance je určený pohled `*Category*Management`. Tato stránka je přístupná pouze uživatelům s dostatečným oprávněním. Pod základními informacemi o akci obsahuje menu přepínající mezi jednotlivými oblastmi správy. Toto menu umožňuje zobrazit popis akce, seznam účastníků, seznam vedoucích, generování PDF seznamů a formulář pro odeslání emailu účastníkům. Přepínání mezi záložkami využívá prvek *Pills* knihovny Bootstrap[1] a funguje pomocí jQuery[2] a technologie AJAX[3]. Načítání stránky je tak náročnější, ale výhodou je poté plynulá práce s kompletní správou akce.

Při prohlížení seznamu účastníků či vedoucích lze vybrat konkrétního uživatele ze seznamu či vyhledat jakéhokoli uživatele z databáze. Karta zvoleného uživatele se poté zobrazí nad seznamem. Kartu lze kdykoli skrýt kliknutím na tlačítko v pravém horním rohu. Karta v horní části obsahuje základní údaje o uživateli, v prostřední údaje o jeho případné přihlášce na aktuální akci a u spodního okraje tlačítka reprezentující operace, které lze se zvoleným uživatelem provést. Tyto možnosti závisí na podmínce, zda je uživatel na danou akci přihlášen a na případném stavu této přihlášky. Nepřihlášeného uživatele lze přihlásit do role vedoucího či účastníka. Pokud to akce vyžaduje, je při přihlášení zobrazen formulář

[1] Knihovna pro HTML, CSS a Javascript, viz kapitola 4.8
[2] Javascriptová knihovna, viz kapitola 4.7
[3] *Asynchronous Javascript and XML*, viz kapitola 4.9

s dodatečnými údaji pro danou akci. u uživatele přihlášeného na akci jsou za standardních podmínek možné změny stavu reprezentované grafem na obrázku 5.2. Tyto možnosti změny stavu jsou tedy nabídnuty zaměstnanci ve formě tlačítek. Běžně používané změny stavu jsou tak velmi rychle dostupné.



Obrázek 5.2: Graf znázorňující běžné změny stavu přihlášky nabízené zaměstnanci v kartě uživatele a barevné zvýraznění odpovídající stavům. Přerušovaná čára značí akci prováděnou účastníkem. Plná čára možnou reakci zaměstnance.

5.2.6 Členství v SKM

Úkolem presenteru nazvaného `Member` je správa členství v SKM u jednotlivých uživatelů. Presenter obsahuje metody sloužící pro vykreslení vyhledávacích formulářů a metody sloužící pro zaznamenání nového členství, předčasné ukončení již existujícího nebo odstranění členství z databáze v případě například chybného kliknutí při vytváření.

Pohledy jsou v této oblasti dva. Hlavní stránkou je správa členů, která je rozdělena na tři záložky. Všechny tři využívají pro načtení a aktualizaci seznamů technologii AJAX[4].

Úvodní záložkou je zobrazení členů za daný školní rok. Automaticky je zobrazen aktuální rok, ale výběrem z nabídky lze zvolit jakýkoli jiný školní rok zahrnutý v databázi. Výběr probíhá ze seznamu, který je vytvořen dynamicky a zahrnuje školní roky od nejstaršího záznamu v databázi po aktuální školní rok.

Druhá záložka slouží pro vyhledávání uživatelů, kteří jsou během zvoleného období přihlášení na minimálně dvě akce a zároveň nejsou členy[5]. Automaticky přednastavené jsou dny ohraničující aktuální školní rok, ale lze zadat jakékoli datum. Po vyhledání jsou zobrazeni všichni uživatelé, kteří splňují dané parametry. v přehledu jsou kromě jejich jména a věku zobrazeny také akce, na které je uživatel v zadaném období přihlášen.

Poslední záložka obsahuje vyhledávání v databázi uživatelů. Vybraný uživatel je zobrazen ve druhém pohledu, který spadá pod `MemberPresenter`. Tento pohled má za úkol

[4] *Asynchronous Javascript and XML*, viz kapitola 4.9
[5] Pravidla kdy je členství vyžadováno byla popsána v kapitole 3.2

zobrazit základní údaje o uživateli a historii jeho členství. Na stránce se nachází také formulář sloužící k evidování nového členství daného uživatele. k tomu je potřeba zadat den přihlášení a školní rok, pro který přihláška platí. Nabídnuty jsou automaticky aktuální datum a probíhající školní rok. v tabulce obsahující existující členství lze daný záznam označit za ukončený či úplně odstranit z databáze pokud byl například vytvořen nesprávně.

5.2.7 Generování seznamů

Další funkcí aplikace je generování seznamů účastníků. Samotné generování probíhá v metodě spadající pod `EventBasePresenter`. Obsah výsledného dokumentu je uživatelsky nastavitelný a `ListPresenter` má na starosti správu definovaných seznamů. Třída obsahuje definice proměnných, ve kterých jsou uloženy možné uživatelské volby definující obsah a formátování výsledné tabulky, a metody pro vytvoření či zpracování formulářů, které se seznamy, tabulkami či sloupci v nich pracují.

Do této části aplikace spadá jeden hlavní pohled, který zobrazuje přehled vytvořených seznamů. u každého z nich je zobrazen název, popis a přehled tabulek, které pod něj spadají. u těchto tabulek je schematicky naznačeno záhlaví tabulky zobrazující sloupce, které tabulka obsahuje, a na místě obsahu tabulky se nachází její název a přehled stavů přihlášek, které jsou do tabulky zahrnuty.

Další pohledy umožňují vytváření a editaci seznamů či tabulek. Vytvoření seznamu obnáší pouze určení názvu a popisu. k existujícímu seznamu lze poté přiřadit libovolné množství tabulek. Při vytvoření tabulky jsou vyžadovány údaje o tabulce a sloupcích. Mezi údaje o tabulce patří název a výběr stavů přihlášek, které jsou v seznamu zahrnuty. Sloupců zobrazených v tabulce může být opět libovolné množství. Formulář je proto nutné vytvářet dynamicky a uživatel může během jeho vyplňování libovolně přidávat či ubírat počet sloupců. Podle počtu sloupců je poté pro každý zobrazena skupina formulářových prvků sloužící pro vyplnění textu zobrazeného jako záhlaví sloupce, dále obsah sloupce, který je vybírán z připravené nabídky, zarovnání sloupce a případné formátování dat. Titulek sloupce si může uživatel zvolit libovolný vepsáním do textového pole. Zbývající hodnoty vybírá z připraveného seznamu. Tyto hodnoty jsou definovány ve třídě `ListPresenter`. Jako datovou hodnotu obsahu sloupce si uživatel může zvolit z dat uchovávaných o uživatelích či přihláškách v databázi. Pro zarovnání jsou dostupné možnosti *Doleva*, *Na střed* a *Doprava*. Třetí seznam volitelně umožňuje zvýraznit hodnoty v buňce barevně podle pohlaví uživatele nebo podle stavu jeho přihlášky.

5.2.8 Odesílání elektronické pošty

Důležitým prvkem usnadňujícím komunikaci s účastníky akcí je možnost odeslání hromadné zprávy uživatelům přihlášeným na konkrétní akci. Formulář pro vytvoření zprávy je definován ve třídě `EventBasePresenter`. Ve stejné podobě je použit u všech typů akcí.

Emailové adresy příjemců emailu lze zadat do textového pole. Jednotlivé adresy jsou odděleny čárkou. Jako výchozí hodnota jsou v tomto poli připraveny automaticky emailové adresy všech účastníků se schválenou přihláškou. Adresy jiné často používané skupiny příjemců lze vyplnit automaticky po kliknutí na tlačítko v záhlaví formuláře. Tento seznam adres lze poté libovolně editovat, smazat některé emaily nebo naopak doplnit další adresáty. Při pokusu o odeslání emailu dojde k ověření platnosti adres. Pokud se nepodaří adresy načíst nebo některé z nich nejsou pomocí PHP funkce `filter_var` vyhodnoceny jako platné, jsou chybné adresy vypsány v chybové hlášce.

Další tři pole obsahují předmět, přílohy, text emailu a podpis, který je připojen na konec textu. Pole pro předmět je automaticky vyplněno názvem akce a pole pro podpis jménem, příjmením a emailovou adresou přihlášeného uživatele. Obojí lze libovolně editovat.

Formulář navíc obsahuje možnost vložit do emailu odkaz na odpovídající akci. Pokud je toto pole při odeslání zatržené, do emailu je vložen blok s informací o akci, od které byla zpráva odeslána. Součástí tohoto bloku je odkaz směřující na detail akce v přihlašovacím systému.

Druhým typem emailu odesílaného informačním systémem je odkaz na obnovení hesla. Tyto zprávy jsou generovány automaticky a obsahují kromě doprovodného textu také odkaz na obnovení hesla v informačním systému. Postup obnovy hesla je podrobněji popsán v kapitole 5.4.1.

5.2.9 Soubory

U akcí je často potřeba účastníkům dát k dispozici nějaký soubor. Může se jednat například o potvrzení od lékaře nebo dokument vyžadující podpis zákonného zástupce. Správu takových souborů zajišťuje `FilePresenter`. Soubory používané v systému jsou nejprve nahrány na server a evidovány v databázi, kde je zvoleno jejich jméno a popis. Tím vzniká takzvaná *knihovna souborů* obsahující informace o všech souborech použitých v rámci informačního systému. Výhodou je poté jednoduché opakované využití souborů a předcházení uložení stejného souboru vícekrát. Všechny akce využívající stejný soubor odkazují na stejný záznam v databázi. v případě změny u souboru lze vyměnit soubor či aktualizovat jeho popis a změna se projeví okamžitě u všech akcí, které daný soubor využívají.

5.2.10 Dárci

Evidence dárců využívá části databáze oddělené od zbytku aplikace. Samotní dárci od databáze přístup nemají a jsou uloženi ve vlastní tabulce. Základní pohled je rozdělen na tři záložky. První z nich zobrazuje přehled darů, který lze omezit na zvolené období. Další dvě obsahují tabulky se seznamem osob a firem. Nového dárce lze vytvořit pomocí formuláře, který se vyskytuje nad tabulkou a zobrazí se po kliknutí na tlačítko. Díky tomu uživatel přehledně vidí, zda již dárce v seznamu existuje, a případně jej může bez přepínání obrazovek vytvořit. u existujícího dárce se lze přepnout na jeho detail. Ten obsahuje informace o dárci a seznam jeho darů. Nový dar lze opět přidat pomocí formuláře, který je na stránce skrytý a objeví se po kliknutí na příslušné tlačítko.

`DonorPresenter` definuje položky formulářů pro vytvoření dárce či daru. Před zobrazením pohledu vždy načítá data potřebná pro zobrazení tabulky darů i dárců a připraví proměnné `selectedDonationInfo` a `selectedDonationDonor`. Ty jsou v inicializovány na hodnotu `null` a za běžných okolností je odpovídající blok skryt. Po výběru konkrétního daru jsou zmiňované proměnné naplněny pomocí technologie AJAX[6] a na stránce se zobrazí blok s podrobnostmi o daru.

5.3 Komponenty

Jedná se o prvky, které lze v aplikaci použít na více místech. Jsou uložené ve složce `app\components`. Ve své aplikaci jsem je využíval pro vykreslení seznamů. Jedná se například o seznam vedoucích, kterému se jako parametr předá seznam přihlášek v roli vedoucího

[6] *Asynchronous Javascript and XML*, viz kapitola 4.9

a komponenta vykreslí tabulku vedoucích a jejich kontaktních údajů. Druhé typické použití je pro zobrazení informací o akci, které se vyskytuje v aplikaci na mnoha místech. Je tak výhodné mít je definované pouze jednou a poté z různých pohledů volat opakovaně komponentu s předanými údaji o akci.

Příklad komponenty v aplikaci

Příkladem komponenty v této aplikaci je karta akce. Tato karta obsahuje základní informace o akci, mezi které patří název, datum konání a popis. Podobná karta je v systému zobrazována na mnoha místech. o její vykreslení se proto stará komponenta nazvaná `EventBlockControl`. Definovaná je ve dvou souborech. Jedním je soubor s příponou `.php`, starající se o vytvoření odpovídající třídy. Ta obsahuje metodu render, která může přijímat parametry a ve které jsou připraveny proměnné pro šablonu. Tato šablona je obsažena v souboru stejného názvu s příponou `.latte`. v této šabloně je popsán vzhled výsledné karty akce.

5.4 Uživatelské účty

Aplikace umožňuje používání následujících úrovní uživatelských oprávnění:

- Běžný uživatel
- Vedoucí
- Ekonom
- Zaměstnanec
- Administrátor

Tyto úrovně jsou seřazeny od té s nejmenšími právy po tu nejvyšší. Každá úroveň vždy dědí všechny práva od předchozí a přidává nové navíc. Požadavky na tyto role a případy užití jsou popsány v kapitole 3.3.2.

5.4.1 Autentizace

Jedná se o ověření při přihlašování uživatele. v Nette frameworku se pro toto ověření využívá třída implementující rozhraní `IAuthenticator`. Toto rozhraní obsahuje jedinou metodu, která se jmenuje `authenticate()`. Tato metoda má za úkol vrátit takzvanou *identitu* uživatele, nebo vyvolat výjimku. Jednoduchá ukázka implementace tohoto rozhraní je vložena do základní struktury Nette projektu.[15]

Ve své aplikaci jsem využil tento základní autentikátor a implementoval vlastní postup pro ověření uživatelského emailu a hesla. Tento postup spočívá v načtení otisku hesla z databáze a ověření shody zadaného hesla s otiskem pomocí PHP funkce `password_verify`. Navíc je po úspěšném ověření provedena kontrola, zda otisk odpovídá požadovaným parametrům. v případě že ne, je do databáze uložen nový otisk. Při každém úspěšném přihlášení je také do databáze uložen datum a čas poslední návštěvy daného uživatele. Další podrobnosti týkající se zabezpečení hesel jsou popsány v kapitole 5.5.

V případě, že uživatel zapomene heslo, může si požádat o jeho obnovu pro účet s konkrétním emailem. v takovém případě je uživateli emailem zaslán odkaz na stránku pro obnovou hesla s náhodně vygenerovaným řetězcem. Otisk řetězce společně s aktuálním časem

je uložen do databáze. Pokud uživatel přistoupí na nabídnutý odkaz, je ověřena správnost bezpečnostního řetězce pro daný email a jeho platnost. Doba platnosti je definována ve třídě `BasePresenter`. v aktuální implementaci se jedná o 30 minut. v případě že je řetězec platný, zobrazí se uživateli formulář pro vytvoření nového hesla.

5.4.2 Autorizace

Autorizace zajišťuje ověření, zda přihlášený uživatel má oprávnění pro přístup na zvolenou stránku či provedení konkrétní akce. Ve své aplikaci jsem použil uživatelské účty popsané v kapitole 5.4. Každý uživatel může zastávat pouze jednu z těchto rolí a role jsou uspořádané tak, že nadřazená vždy disponuje všemi právy podřízené role.

O autorizaci se v Nette frameworku stará třída implementující rozhraní `IAuthorizator`. Lze samozřejmě opět definovat tuto třídu manuálně, ale pro potřeby této práce to nebylo nutné. Využil jsem tedy připravenou třídu a v konfiguračním souboru `config.neon` nastavil specifikaci tzv. *Access Control Listu*. Práce s ním vyžaduje definici rolí, zdrojů a oprávnění.[15]

Role v této aplikaci jsou *Uživatel*, *Vedoucí*, *Ekonom*, *Zaměstnanec* a *Administrátor*. Zdroje jsem rozdělil do dvou typů. Jedná se o zdroje odpovídající entitám a zdroje odpovídající pohledům. Tyto entity a stupně oprávnění jsou následující:

- Akce
  - Přihlásit se na akci
  - Vyhledat akce v databázi
  - Upravit údaje o akci
  - Smazat akci
- Kategorie akcí
  - Upravit popis
- Uživatel
  - Zobrazit kontaktní údaje
  - Zobrazit historii uživatele
  - Zobrazit podrobnosti o účtu
  - Vyhledávat uživatele v databázi
  - Upravit uživatelské údaje
  - Vymazat údaje o uživateli z databáze
- Seznam přihlášených
  - Zobrazit přijaté přihlášky
  - Zobrazit přihlášky ve všech stavech
  - Zobrazit čas přihlášení
  - Upravit údaje
- Přihláška

  - Nastavit datum zaplacení
  - Změnit stav přihlášky
- Seznam vedoucích
  - Zobrazit seznam vedoucích
  - Přidat nového vedoucího
- Menu
  - Zobrazit vzdělávací akce
  - Zobrazit sekci pro zaměstnance
- Soubory
  - Spravovat soubory
- PDF seznamy
  - Generovat seznam
  - Spravovat předpisy pro generování
- Emaily
  - Poslat email účastníkům

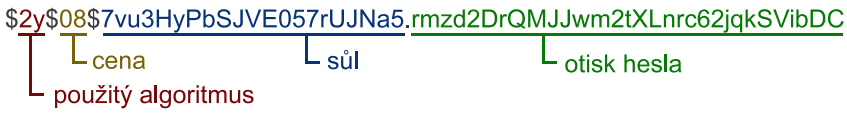
Dalším používaným typem je oprávnění pro přístup k pohledu. Zde není specifikován stupeň a jedná se pouze o ověření, zda uživatel má přístup na stránku nebo mu bude zobrazena chyba s kódem 403. Pohledy s omezeným přístupem jsou následující:

- Vyhledávání akcí v databázi
- Vyhledávání uživatelů v databázi
- Historie uživatele
- Správa akce
- Akce z kategorie *Vzdělávací*
- PDF seznamy
- Knihovna souborů
- Členové SKM
- Dárci

5.5 Zabezpečení

Pro ukládání uživatelských hesel jsem využil vytváření jejich otisků pomocí PHP funkce `password_hash`. v databázi tedy není uloženo heslo, ale návratová hodnota této funkce. Jedná se o řetězec znázorněný na obrázku 5.3.

Pro vyšší bezpečnost se k heslu před vytvořením otisku přidává řetězec složený z náhodných znaků. Tento řetězec se nazývá *sůl* (slovo je i v českém prostředí často používané v anglickém originále *salt*). Použití soli má za cíl znesnadnit případnému útočníkovi odhalení hesel. Pokud by vstupem algoritmu bylo samotné heslo, útočník může porovnávat výstupy algoritmu pro různá hesla s otiskem uloženým v databázi. Toto porovnávání lze dále zjednodušit pomocí předem připravené takzvané *duhové tabulky*, která obsahuje hesla a jejich otisky pro daný algoritmus. Útočník by tak byl schopný jednoduše vyhledat otisk z databáze v tabulce a zjistit jemu odpovídající heslo. *Sůl* je náhodný řetězec, který je připojen k heslu ještě před vytvořením otisku. Díky tomu útočník nemůže využít jednu předem připravenou duhovou tabulku, ale musel by vypočítat speciální tabulku pro každou variantu soli. Při použití dostatečně dlouhé náhodně generované soli by to znamenalo provádění algoritmu pro všechny možnosti hesel a pro každého uživatele zvlášť. Zvolený algoritmus by tedy měl být natolik složitý, aby útočníka odradila časová náročnost tohoto postupu.[12]



Obrázek 5.3: Řetězec obsahující otisk uživatelského hesla a parametry využité pro jeho vytvoření. (převzato z [12])

Výsledkem funkce `password_hash` je řetězec, obsahující údaje o použitém algoritmu, sůl a otisk. Příklad takového řetězce je na obrázku 5.3. Takový řetězec je uložen v databázi a na základě informací v něm obsažených poté funkce `password_verify` dokáže vyhodnotit, zda zadané heslo odpovídá otisku. Podporované algoritmy jsou *bcrypt* a od PHP verze 7.2.0. také *argon2*. Lze zvolit jeden z těchto algoritmů, nebo využít konstantu `PASSWORD_DEFAULT`. v takovém případě bude využit algoritmus, který je pro danou verzi PHP vyhodnocen jako doporučený. Aktuálně se jedná o *bcrypt*, ale v budoucnu by měl být vždy nahrazován algoritmem vyhovujícím svou náročností dostupným výpočetním prostředkům.[13]

Bezpečnost hašovacích funkcí je závislá na aktuální výkonnosti výpočetních technologií. Právě vlivem jejich vývoje se dříve používané algoritmy stávají nedostatečně bezpečnými. Důvodem je příliš nízká časová náročnost výpočtu těchto algoritmů při využití dnes dostupných technologií. Útočník má díky tomu možnost v dostatečně krátkém čase porovnat hodnotu uloženou v databázi s otisky všech možných variant hesla.[12]

5.6 Uživatelské rozhraní

Následující kapitola popisuje prvky uživatelského rozhraní a jejich řešení v aplikaci.

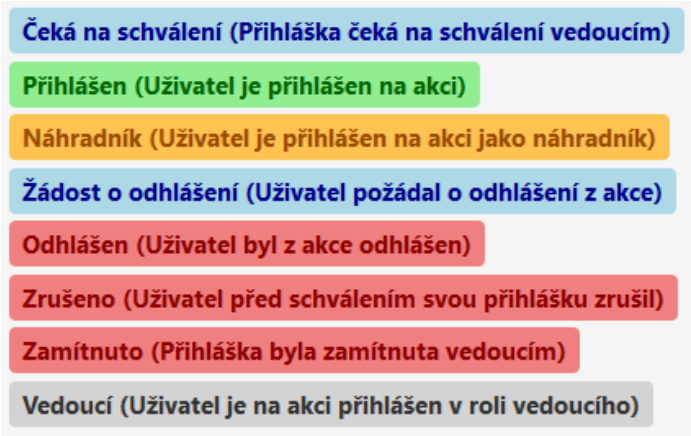
Tlačítka

Pro zvýraznění tlačítek jsem vytvořil CSS předpis v souboru `Buttons.css`. Cílem bylo sjednotit vzhled tlačítek v celé aplikaci. Tyto tlačítka jsou určeny třídou `button`, která je ve specifických případech doplněna další třídou. Při použití této třídy získávají stejný vizuální vzhled jak HTML tlačítka, tak odkazy plnící funkci tlačítka. Tento vzhled by měl tlačítka na první pohled odlišovat tak, aby bylo uživateli zřejmé že se jedná o interaktivní prvky provádějící danou akci[7].

Kromě základní třídy `button` může tlačítko obsahovat také další doplňující třídu, která například specifikuje jeho barvu nebo velikost.

Barvy

Rozlišení pomocí barev jsem použil při zvýraznění aktuálního stavu přihlášky. Jednotlivé stavy jsou popsány v kapitole 3.4.1. Použité barvy jsou definované pomocí kaskádových stylů v souboru `Application.css` a přiřazeny požadovaným objektům na stránce pomocí odpovídající HTML třídy. Použité barvy jsou ilustrovány na obrázku 5.4.



Obrázek 5.4: Možné stavy přihlášky. Barva textu a pozadí u stavů odpovídá barvám použitým v aplikaci.

Použití těchto barev zvyšuje přehlednost tabulky zobrazující účastníky akce. Sloupec obsahující stav přihlášky kromě textu využívá odpovídající formátování. Uživatel tak velmi rychle získá přehled o počtu přihlášek které jsou v pořádku, počtu odhlášených či výskytu přihlášek vyžadujících reakci zaměstnance. Odpovídající barvy jsou využity také pro tlačítka rozhodující o přijetí přihlášky. Kliknutí na zelené tlačítko převede přihlášku do stavu *Přihlášen*, se kterým koresponduje zelená barva. Analogicky jsou použity také žlutá a červená barva signalizující stav *Náhradník* a *Zamítnuto*.

Orientace na stránce

Pro orientaci na stránce je důležité především menu a horní lišta. Tyto dva prvky jsou definovány v hlavní šabloně nazvané v Nette frameworku `@layout.latte` a využity ve stejné podobě na všech stránkách aplikace. Horní lišta obsahuje logo Domu Ignáce Stuchlého a možnosti spravující uživatelský účet. Tyto možnosti jsou pro nepřihlášeného uživatele buď přihlášení existujícím účtem, nebo vytvoření účtu nového. Pro přihlášeného uživatele je možné zobrazit informace o svém účtu nebo se odhlásit. Menu obsahuje odkazy na jednotlivé

typy akcí a pro uživatele s vyšším oprávněním také barevně oddělenou část menu s dalšími povolenými pohledy.

Tyto prvky jsou pro uživatele důležitým záchytným bodem. Logo v levém horním rohu stránky plní především dvě funkce. Sděluje návštěvníkovi, na jaké webové stránce se nachází a při kliknutí zobrazí hlavní stránku aplikace. Tato možnost rychlého návratu na hlavní stránku zvyšuje uživatelský komfort. Návštěvník díky tomu může kdykoli provést rychlý návrat zpět pokud se například omylem dostane na stránku, kterou navštívit nechtěl.[7]

Dalším prvkem je vizuální odlišení aktuálně zvolené položky v menu. Toho je využito jak v hlavním menu, tak také u záložek využitých na stránkách s velkým množstvím obsahu. u stránek zobrazujících jednotlivé typy akcí je také využito naznačení cesty pomocí takzvané drobečkové navigace[7]. Ta spočívá v zobrazení cesty od nadřazené položky k podřazeným, přičemž nadřazená vždy slouží jako odkaz na odpovídající stránku[7]. v konkrétním případě na obrázku 5.5 je zobrazen titulek stránky *Příměstské tábory* spadající pod kategorii *Tábory*.

Obrázek 5.5: Praktické využití drobečkové navigace v popisované aplikaci.

[7] anglicky *breadcrumb navigation*

Kapitola 6

Testování

Tato kapitola popisuje testování uživatelského rozhraní výsledné aplikace. Je rozdělena na tři základní části. Podkapitola 6.1 obsahuje návrh způsobu testování a prováděné úkony. Následuje část 6.2 popisující průběh testování a dosažené výsledky. v závěrečné části 6.3 se nachází vyhodnocení testů a popis změn provedených v aplikaci.

6.1 Návrh uživatelských testů

Pro testování uživatelského rozhraní jsem zvolil individuální pozorování uživatelů se strukturovaným seznamem úkolů a otázek. Jedná se o časově náročnou metodu, která ale umožňuje získání podrobnějších dat. Při osobním kontaktu mají uživatelé tendenci sdělit více informací a obvykle je pro ně příjemnější než komunikace na dálku či vyplňování dotazníku.[14]

Aplikace vytvářená v rámci této bakalářské práce obsahuje velké množství funkcí, ale většina z nich je přístupná pouze zaměstnancům Domu Ignáce Stuchlého. Nejnáročnější část návrhu uživatelského rozhraní se týká správy akcí pro zaměstnance. Ta umožňuje velké množství úkonů, které lze s danou akcí provést. Jedná se ale o část aplikace přístupnou pouze úzké skupině uživatelů. Proto jsem jako nejvhodnější způsob testování zvolil právě individuální setkání se zaměstnanci.

Testování spočívalo v provádění zadaných úkolů. Tyto úkoly jsem zadával ústně. Statistiky sbírané během plnění úkolu jsou následující:

- Čas potřebný pro dokončení úkolu
- Počet návratů o krok zpět nebo zbytečných kliknutí, které neměly vliv na plnění úkolu
- Počet návratů na úvodní stránku aplikace nebo na stránku ze které bylo plnění úkolu zahájeno
- Počet situací, kdy uživatel nakonec zvolil správný postup, ale dal najevo nejistotu nebo mu rozhodování trvalo příliš dlouho
- Počet situací, kdy uživatel nebyl schopný úkol vyřešit bez nápovědy
- Subjektivní hodnocení obtížnosti úkolu na stupnici od 1 do 5. Hodnota 1 znamená jednoduchý úkon nevyžadující přemýšlení. Číslo 5 vyjadřuje náročný úkol jen těžce splnitelný bez nápovědy.

- Hodnota na stupnici od 1 do 5 vyjadřující frekvenci provádění odpovídající činnosti v praxi. Číslo 1 odpovídá úkonu prováděném jednou za dlouhý čas a 5 vyjadřuje náplň každodenní práce.

Po splnění úkolu uživatel na stupnici od 1 do 5 vyjádřil svůj osobní názor na náročnost daného úkolu a, pokud se jednalo o zaměstnance, také frekvenci provádění této činnosti v praxi. Za každým úkolem poté následoval prostor pro podrobnější slovní hodnocení a nápady na zlepšení aplikace.

Následující seznam uvádí úkoly zadávané uživatelům:

1. Přihlášení do aplikace účtem Jana Kopeckého (běžný uživatel) a odeslání přihlášky na pobytový tábor "Z pohádky do pohádky"
2. Podání žádosti o odhlášení z akce "Sportovní kemp" a stažení souboru "Návod k pračce", který je u ní uložený
3. Vyhledání, které akce se přihlášený uživatel v minulosti zúčastnil a změna dne narození uživatele

Zbývající úkoly se týkají tábora "Z pohádky do pohádky"

1. Přihlášení do aplikace účtem Petr Novák (zaměstnanec) a potvrzení odhlášení Jana Kopeckého z tábora "Sportovní kemp"
2. Označení Jana Kopeckého jako náhradníka a schválení ostatních žádostí o přihlášení
3. Zapsání Martina Hofmana na akci v roli vedoucího
4. Přidání nového souboru z knihovny uložené v systému
5. Odeslání zprávy, která bude adresovaná všem účastníkům s aktuálně schválenou přihláškou. Obsahem má být připomínka, aby si účastníci nezapomněli svačinu a ze zprávy musí být zřejmé které akce se týká
6. Vygenerování seznamu účastníků akce ve formátu PDF. Seznam má zahrnovat účastníky včetně náhradníků a zobrazovat sloupce jméno (s barvou textu podle pohlaví účastníka), příjmení a email. *Zadanou podobu neměl žádný předdefinovaný seznam a bylo tak nutné vytvořit vlastní předpis.*

První tři úkoly se zaměřují na používání systému běžným uživatelem. Úkoly číslo 4 a 5 reagují z pohledu zaměstnance na žádosti vytvořené v rámci úkolů 1 a 2. Díky tomu měl testující uživatel příležitost lépe pochopit souvislosti a provázání těchto událostí. Zbývající úkoly poté využívají dalších možností dostupných pro zaměstnance ze stránky konkrétní akce.

Aplikace dále obsahuje pohledy obsluhující správu dárců a správu členství uživatelů v SKM[1]. Tyto pohledy bude nejčastěji využívat jediný zaměstnanec. u následujících úkolů zaměřených na zmiňované oblasti jsem tak přikládal největší význam testování konkrétního zaměstnance s odpovídající kompetencí.

7. Zobrazení seznamu členů SKM za aktuální rok a vložení členství Antonína Říhy platného od aktuálního dne.
8. Zaevidování daru od firmy "ABCD', která ještě v minulosti nepřispěla.

[1]Salesiánské kluby mládeže, více informací o členství je popsáno například v kapitole 3.2

6.2 Průběh a výsledky

Samotné testování proběhlo ve dvou fázích. Nejprve přímo v domě Ignáce Stuchlého se třemi zaměstnanci. Následně jsem provedl drobné úpravy týkající se převážně názvů tlačítek, které se v první části testování jevily jako nejasné. Ve druhé fázi poté testování prováděli tři noví uživatelé, kteří se s aplikací předem nesetkali. Ti nepatří mezi zaměstnance Domu Ignáce Stuchlého, ale jsou seznámeni s jeho fungováním. Zadání pro ně bylo stejné. Jediný rozdíl ve sledování výsledků byl u otázky na frekvenci provádění daného úkonu v praxi. Na tuto otázku odpovídali vždy jen zaměstnanci, kteří daný úkol provádí v rámci svých pracovních kompetencí.

Výsledky testování jsou shrnuté v následujících tabulkách. Tabulka 6.1 zobrazují shrnutí výsledků pro tři úkoly prováděné v roli běžného uživatele, tabulka 6.2 zobrazuje výsledky úkolů prováděných standardně všemi zaměstnanci a tabulka 6.3 zobrazuje úkoly prováděné obvykle pouze administrativním pracovníkem.

Sloupce tabulky odpovídají jednotlivým úkolům. Řádky odpovídají pro daný úkol agregaci naměřených výsledků u všech uživatelů. Znění jednotlivých úkolů a podrobnější popis sbíraných dat je v kapitole 6.1.

Tabulka 6.1: Výsledky testování uživatelského rozhraní aplikace z pohledu běžného uživatele.

Číslo úkolu	1	2	3
Průměrný čas	01:36	00:56	01:40
Kroků zpět	2	0	0
Návratů na úvodní stránku	0	0	1
Nejistých kroků	1	1	1
Nápověd	0	0	0
Průměrná subjektivní obtížnost	1,17	1,00	1,08
Průměrná frekvence používání	3,67	2,67	2,00

Výsledky testování části přihlašovacího systému dostupné pro běžné uživatele dopadla velmi dobře. u všech úkolů vždy pět uživatelů zvládlo vše bez komplikací na první pokus a jeden uživatel s drobnou nejistotou, ale samostatně. Také hodnocení obtížnosti úkolů bylo velmi nízké. Návrhy na zlepšení se týkaly přehlednosti stránky požadující vyplnění údajů specifických pro danou akci a lepší rozlišení pojmů *přihlášení na akci* a *přihlášení do systému*.

Tabulka 6.2: Výsledky testování uživatelského rozhraní aplikace z pohledu zaměstnance zodpovědného za akci.

Číslo úkolu	4	5	6	7	8	9
Průměrný čas	01:49	01:21	03:02	00:22	02:36	07:09
Kroků zpět	2	2	3	0	3	6
Návratů na úvodní stránku	1	0	5	0	2	1
Nejistých kroků	4	3	3	0	3	3
Nápověd	0	1	1	0	1	0
Průměrná subjektivní obtížnost	2,00	2,75	2,20	1,00	1,80	3,30
Průměrná frekvence používání	4,33	3,50	3,50	4,00	3,50	1,50

Úkony prováděné běžně zaměstnancem zodpovídajícím za konkrétní akci jsou hlavní částí systému. Stránka zajišťující správu akce také obsahuje pro uživatele nejvíce možností. Přístup k ní mají pouze zaměstnanci Domu Ignáce Stuchlého. Výsledky testování byly v této oblasti nejrozmanitější. Nejlepší hodnocení získal úkol číslo 7, u kterého nedošlo k žádným měřitelným komplikacím. Navíc se jedná o často prováděný úkon.

Významnou skupinou jsou úkoly 4, 5 a 6. Ty pracují s přihláškami účastníků. Subjektivní hodnocení obtížnosti u nich je průměrné a čas svou délkou nevyčnívá. Objevuje se zde ale větší množství chybných a nejistých kroků a ve dvou případech také nutnost slovní nápovědy. Jako důvod jsem vyhodnotil především dvě věci. První bylo nejasné pojmenování tlačítka pro zobrazení detailu uživatele a jeho přihlášky na akci. Druhým důvodem bylo oddělení vyhledávání uživatelů z databáze na samostatnou záložku. Uživatelé vždy dlouho přemýšleli jak zareagovat na jednu konkrétní žádost. Také měli problém zjistit že pro přihlášení účastníka či vedoucího potřebují záložku vyhledávání uživatelů. Jako řešení jsem provedl následující změny:

- Změna názvu bloku obsahujícím informace o uživateli a jeho přihlášce na *kartu uživatele*
- Odpovídající změna textu na tlačítku zobrazujícím tuto kartu
- Rozšíření popisu v záhlaví záložky. Vyjmenování možností, které karta obsahuje, a postup pro její zobrazení
- Přesunutí vyhledávání v databázi z oddělené záložky přímo k seznamu účastníků i vedoucích

Pozitivně naopak uživatelé hodnotili použití barev pro zvýraznění stavu přihlášky u jednotlivých uživatelů.

Poměrně nízké hodnocení i přes velké množství chybných kliknutí získalo odeslání emailu účastníkům. Samotné vytvoření emailu bylo obvykle bez komplikací, ale jeho otevření nebylo pro všechny uživatele intuitivní. Formulář je umístěn u akce na vlastní záložce a část uživatelů jej hledala na stránce se seznamem uživatelů. Jako řešení jsem zvolil umístění odkazu na záložku s formulářem k tabulce se seznamem účastníků.

Posledním úkolem bylo vytvoření vlastního předpisu pro generovaný seznam účastníků. Tento úkol byl časově nejnáročnější a byl také jako nejnáročnější hodnocen, což odpovídalo očekáváním. Jedná se o úkon, který nebude prováděn příliš často. Předpokladem je, že ve většině případů budou zaměstnancům postačovat již předem definované předpisy pro generování seznamu. Krokem pro zlepšení uživatelské přívětivosti aplikace bylo rozšíření popisu v záhlaví odpovídající záložky. Text byl upraven, aby lépe popisoval princip, jakým generováním seznamů funguje.

Tabulka 6.3: Výsledky testování uživatelského rozhraní aplikace z pohledu administrativního pracovníka.

Číslo úkolu	10	11
Průměrný čas	01:26	01:50
Kroků zpět	0	0
Návratů na úvodní stránku	0	0
Nejistých kroků	1	0
Nápověd	0	0
Průměrná subjektivní obtížnost	1,00	1,00
Průměrná frekvence používání	4,00	3,00

Zbývající dvě části aplikace se týkají evidence dárců a členství v SKM. Obě jsou méně komplexní než části testované v předchozích úkolech. Tomu odpovídá i hodnocení, které je velmi dobré.

6.3 Vyhodnocení

Celkově výsledky testování hodnotím jako velmi dobré. Intuitivnost a jednoduchost je důležitá především pro běžné uživatele, kteří se systémem budou pracovat výjimečně. Testování této části aplikace přineslo výborné výsledky a zapotřebí byly jen drobné změny.

Nejsložitější částí aplikace je správa akce, která umožňuje velké množství úkonů. Zde testování odhalilo několik překážek v uživatelské přívětivosti aplikace. Většina problémů se opakovala u více testujících uživatelů a na základě doporučení získaných při testování proběhly v této části nejvýraznější změny. Přístup ke správě akce má pouze úzká skupina zaměstnanců, která bude se systémem pracovat pravidelně. Při nasazení aplikace do ostrého provozu proto může komplikacím předejít vytvoření uživatelského manuálu. Zaměstnanci tak budou mít příležitost během seznamování s aplikací získat při provádění potřebných úkonů rady a naučit se jednoduchý postup řešení.

Kapitola 7

Zhodnocení výsledku

Během tvorby databáze a aplikace se povedlo splnit všechny požadavky organizace popsané v kapitole 3.3. Testování uživatelského rozhraní potvrdilo použitelnost výsledného systému, který byl na základě výsledků testování dále vylepšen. Systém nabízí zaměstnancům velké množství funkcí, které zjednodušují činnosti prováděné v současné době zdlouhavým a komplikovaným způsobem. Prostor pro další zlepšení vidím v přehlednosti některých konkrétních částí aplikace.

Jde například o generovaní seznamů a vytváření jejich předpisů, které probíhá prostřednictvím formuláře. Pro lepší přehled by bylo ideální znázornit uživateli vzhled aktuálně vytvářené tabulky. Případně také umožnit editaci tabulky přímo klikáním na její buňky. Jedná se ale o činnost, u které se nepředpokládá časté provádění a během testování ji všichni uživatelé zvládli bez nápovědy.

Dalším takovým prvkem je vyplnění adresátů emailu rozeslaného účastníkům akce. Zde si opět dokážu představit uživatelsky přívětivější rozhraní s možností například rychle odebrat adresáta kliknutím na jedno tlačítko. Zároveň by bylo vhodné zobrazit u každého adresáta kromě emailu i jméno a příjmení. Tyto změny by pomohly zaměstnancům v situacích, kdy potřebují zprávu poslat pouze specifické skupině účastníků. Nejčastěji používané skupiny (například všechny účastníky se schválenou přihláškou) lze zvolit jako adresáty kliknutím na odpovídající tlačítko v záhlaví formuláře.

7.1 Možné rozšíření

Za další vhodné rozšíření považuji lepší provázání systému s papírovými přihláškami vyžadovanými pro účast na akci. v současné době systém umožňuje nahrát k akci soubor a nabídnout jej účastníkům ke stažení. Pokud by předpis popisující univerzální vzhled papírové přihlášky bylo možné uložit do databáze, dokázal by systém využít data známá o jednotlivých uživatelích a každému vytvořit personalizovanou přihlášku. Počet položek vypisovaných ručně by se v ideálním případě zúžil pouze na datum a podpis účastníka či zákonného zástupce.

Velkým rozšířením by v budoucnu mohlo být také zahrnutí Orientačních dnů® do jednotného systému. Ty jsou popsány v kapitole 3.1. Přihlašovaní na ně má mnoho specifik a vyžadovalo by vlastní část aplikace pracující s oddělenými tabulkami v databázi.

Kapitola 8

Závěr

Cílem této práce bylo vytvořit přihlašovací systém pro Dům Ignáce Stuchlého ve Fryštáku. Tento systém účastníkům umožní prohlížení plánovaných akcí a přihlašování na ně pomocí webové aplikace. Zaměstnancům má aplikace s využitím dat v databázi usnadnit a zefektivnit práci.

Požadavky organizace se podařilo splnit. Výsledná aplikace umožňuje provádění všech úkonů popsaných v kapitole 3.3. Výsledky testování aplikace z pohledu běžného uživatele popsané v kapitole 6 jsou výborné a uživatelské rozhraní je hodnoceno jako dobře pochopitelné. Výrazné zjednodušení práce zaměstnanců vnímám především v oblasti evidence účastníků jednotlivých akcí. Každý zaměstnanec může mít v rámci systému přístup k seznamu účastníků konkrétní akce. Pomocí dvou kliknutí si poté lze vygenerovat seznam obsahující potřebné informace. Časově velmi náročnou činností je kontrola počtu akcí, na které jsou uživatelé přihlášení během jednoho školního roku. Tato činnost je s využitím databáze automatizována, což šetří čas a předchází chybám.

Práce na této aplikaci mi výrazně rozšířila přehled v oblasti návrhu a implementace informačních systémů. v průběhu bakalářského studia jsem se setkal s mnoha zajímavými předměty, ale všechny byly úzce zaměřené. Vytvářené projekty obvykle vypracovávaly týmy studentů. v této práci pro mě bylo velkým přínosem provedení všech dílčích činností samostatně.

Rozvoj přihlašovacího systému by dále mohl pokračovat implementací užšího propojení systému s papírovými přihláškami, které by dále ušetřilo čas zaměstnancům Domu Ignáce Stuchlého i účastníkům akcí. v budoucnu je také možné pracovat na vytvoření části systému spravující přihlašování škol na Orientační dny®[1].

[1] https://www.disfrystak.cz/orientacni-dny/

Literatura

[1] Baše, O.: *jQuery pro neprogramátory : průvodce využitím knihovny jQuery UI*. Brno: Computer Press, první vydání, 2012, ISBN 978-80-251-3750-5.

[2] *Bootstrap · The most popular HTML, CSS, and JS library in the world.* [Online; navštíveno 28.04.2019].
URL https://getbootstrap.com/

[3] Conolly, T.: *Mistrovství - databáze : profesionální průvodce tvorbou efektivních databází*. Brno: Computer Press, vyd. 1. vydání, 2009, ISBN 978-80-251-2328-7.

[4] *Database Explorer*. [Online; navštíveno 28.04.2019].
URL https://doc.nette.org/database-explorer

[5] Hernandez, M. J. M. J.: *Návrh databází*. Profesional, Praha: Grada, první vydání, 2006, ISBN 80-247-0900-7.

[6] Kosek, J.: *HTML : tvorba dokonalých www stránek : podrobný průvodce*. Průvodce (Grada), Praha: Grada, vyd. 1. vydání, 1998, ISBN 80-7169-608-0.

[7] Krug, S.: *Don't make me think, revisited : a common sense approach to web usability*. San Francisco: New Riders, 2014, ISBN 978-0321965516.

[8] Lacko, L.: *Ajax : hotová řešení*. K okamžitému použití, Brno: Computer Press, vyd. 1. vydání, 2008, ISBN 978-80-251-2108-5.

[9] *Latte*. [Online; navštíveno 28.04.2019].
URL https://latte.nette.org/guide

[10] *MVC aplikace & presentery*. [Online; navštíveno 29.04.2019].
URL https://doc.nette.org/2.4/presenters

[11] Pearce, R.: *Dead database walking: MySQL's creator on why the future belongs to MariaDB*. [Online; navštíveno 28.04.2019].
URL https://www.computerworld.com.au/article/457551/dead_database_walking_mysql_creator_why_future_belongs_mariadb

[12] *PHP: Password Hashing*. [Online; navštíveno 01.02.2019].
URL https://php.net/manual/en/faq.passwords.php

[13] *PHP: password_hash - Manual*. [Online; navštíveno 02.02.2019].
URL https://www.php.net/manual/en/function.password-hash.php

[14] Preece, J.; Rogers, Y.; Sharp, H.: *Interaction Design: Beyond Human-Computer Interaction*. New York: John Wiley, 2002, ISBN 0-471-49278-7.

[15] *Přihlašování & oprávnění uživatelů*. [Online; navštíveno 29.04.2019].
URL https://doc.nette.org/2.4/access-control

[16] *Seznámení s Nette Frameworkem*. [Online; navštíveno 28.04.2019].
URL https://doc.nette.org/getting-started

[17] Sklar, D.: *PHP 7 : praktický průvodce nejrozšířenějším skriptovacím jazykem pro web*. Brno: Zoner press, vydání první vydání, 2018, ISBN 978-80-7413-363-3.

[18] Sodomka, P.: *Informační systémy v podnikové praxi*. Brno: Computer Press, vyd. 1. vydání, 2006, ISBN 80-251-1200-4.

[19] Ullman, L.: *PHP a MySQL : názorný průvodce tvorbou dynamických WWW stránek*. Brno: Computer Press, vyd. 1. vydání, 2004, ISBN 80-251-0063-4.

[20] Zelinka, P.: *Ignác Stuchlý - Salesiáni Dona Boska*. [Online; navštíveno 27.04.2019].
URL https://www.sdb.cz/osobnosti/ignac-stuchly/

[21] Zendulka, J.: *Databázové systémy : IDS*. Brno: Fakulta informačních technologií, 2006.

[22] Černá, A.; Chromý, J.; Konečná, H.; aj.: *Internetová jazyková příručka – Ústav pro jazyk český Akademie věd ČR, v. v. i.* Centrum zpracování přirozeného jazyka FI MU, [Online; navštíveno 24.04.2019].
URL http://prirucka.ujc.cas.cz/

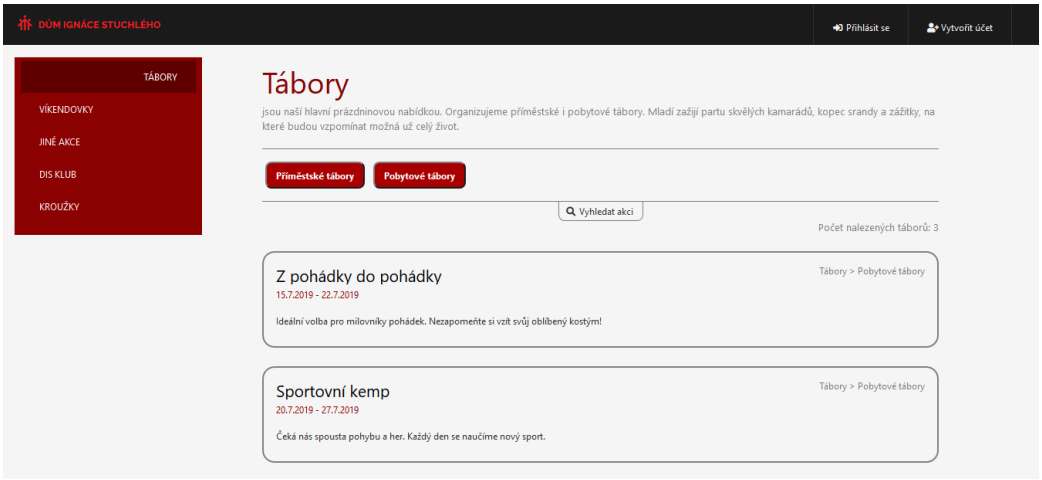
Příloha A

Obsah přiloženého CD

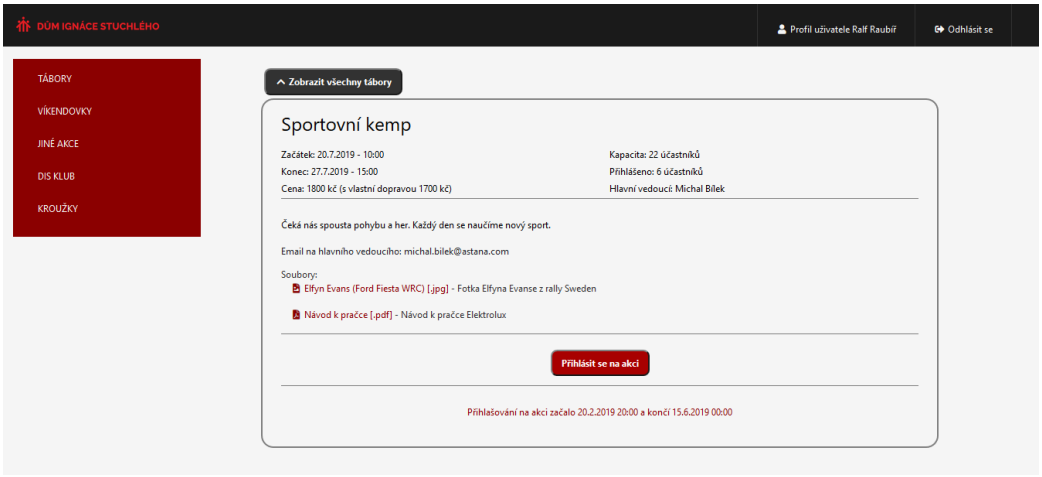
- `xmechl00-BP` - adresář obsahující zdrojové texty aplikace. Jejich struktura je podrobněji popsána v kapitole 5
- `db` - adresář obsahující skripty pro vytvoření databáze a její naplnění ukázkovými daty
- `xmechl00-BP.pdf` - soubor obsahující text práce
- `README.txt` - soubor obsahující instrukce pro spuštění a popisující specifika odevzdané verze aplikace

Příloha B

Ukázka vzhledu aplikace

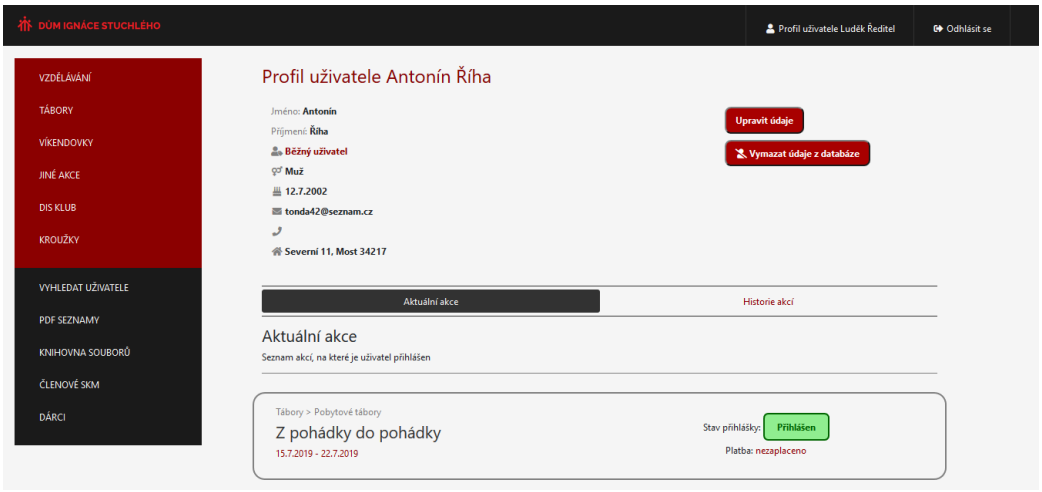


Obrázek B.1: Stránka s přehledem všech akcí ve vybrané kategorii pro nepřihlášeného uživatele



Obrázek B.2: Stránka zobrazující detail akce z pohledu běžného uživatele

Obrázek B.3: Správa akce z pohledu zaměstnance



Obrázek B.4: Zobrazení uživatelského profilu a aktuálních přihlášek vybraného uživatele